Representation Learning-based Approaches for Modeling Data in Multiple Modalities

A Dissertation presented

by

Yingtao Tian

to

The Graduate School

in Partial Fulfillment of the

Requirements

for the Degree of

Doctor of Philosophy

in

Computer Science

Stony Brook University

May 2019

Abstract of the Dissertation

Representation Learning-based Approaches for Modeling Data in Multiple Modalities

by

Yingtao Tian

Doctor of Philosophy

in

Computer Science

Stony Brook University

2019

Computationally modeling the real world starts with modeling the data, which exists in a variety of modalities. Modeling these data enables leveraging rich real-world information for agents to behave more naturally in interaction with humans. However, challenging questions remain on (a) how best to model data from different modalities and (b) how to model them in a unified way for downstream tasks to leverage. The recent successes of representation learning that associates data with continuous vector representations pave directions for a unified way to model data. In this thesis, we address this challenge by studying the approaches in modeling data of multiple modalities using representation learning.

Data from multiple modalities are characterized by structures that call for a variety of techniques. I present a generative model for structured data that captures the representations of discrete structures with formal grammars and semantics, and fast and effective label inference for networks on graphs with multi-labels edges. Another aspect of my work enables generating high-quality facial images of anime characters, a project that has proven popular through the availability of a web interface that runs on edge devices.

Furthermore, I move forward with challenges concerning dealing simultaneously with data from more than one modality. I describe my work modeling bilingual dictionaries, which contains a graph of cross-lingual correspondence between sentences, lexicons, and texts of descriptions to benefit cross-lingual applications such as semantic search and question answering. Finally, I present a novel approach to bridge modularities, by learning a post-hoc interface between two existing models to solve a new task, and facilitate transferring across different modalities and even different types of generative models.

Dedicated to my mother,
who opened the door to computation for me

Contents

List of Figures

List of Tables

Acknowledgements

A Ph.D. journey is adequately considered a marathon rather than a sprint, and mine is not an exception. Yet it has been made a fulfilling life experience for me by many people whom I worked with, to which I would like to express my sincere appreciations.

First, I wish to thank my advisor, Steven Skiena, who mentored me through the whole journey with his continuous support and encouragement for me to strive for a high-standard intellectual pursuit. Moreover, the academic freedom Steven has provided allowed me to explore and make trial-and-err on many topics, which made myself what I have ever thought of. I could not imagine a journal without the generous intellectual and moral support from Steven.

Second, I am honored to have the opportunities collaborating with many wonderful colleges, including lab-mates Haochen, Vivek, Rami, Bryan, Xiaofei, Ruhul, Alisa who have provided me with excellent working experience. Besides, I also enjoyed the working with collaborators from other academic institutes, including Hanjun, Muhao, Vivek, Wenlu and Chunqi and Minju. Moreover, my experience has also been amplified by chances to work in an industry with amazing people including Bryan, Rami, Bo, Jesse, Jiakai, Haixun, Yanghua, and Huachun.

As a very valuable experience, many professional relationships with people have turned into personal friendships beyond merely collaboration. I am grateful to work with Haochen in co-coaching the competitive programming team of Stony Brook University. Also, I enjoyed our (informal) anime fan club with Yanghua, Huachun, Minjun, Jiakai, Chunqi that turned our shared interests in anime into research, papers, some impact on the anime industry, and finally a business.

The names of people I would like to appreciate are many, which this space for acknowledgment is too narrow to contain. I am aware that a list of names can only be incomplete and hopefully I could be forgiven for the incompleteness! Finally, as a general response — I would acknowledge the help from everybody who has helped me and changed me, however significant or slight, through this long journal.

Publications

1. Muhao Chen, Yingtao Tian, Mohan Yang, and Carlo Zaniolo. Multilingual knowledge graph embeddings for cross-lingual knowledge alignment. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI 2017, Melbourne, Australia, August 19-25, 2017*, pages 1511–1517, 2017

2. Yanghua Jin, Jiakai Zhang, Minjun Li, Yingtao Tian, Huachun Zhu, and Zhihao Fang. Towards the automatic anime characters creation with generative adversarial networks. *CoRR*, abs/1708.05509, 2017

3. Muhao Chen, Yingtao Tian, Xuelu Chen, Zijun Xue, and Carlo Zaniolo. On2vec: Embedding-based relation prediction for ontology population. In *Proceedings of the 2018 SIAM International Conference on Data Mining, SDM 2018, May 3-5, 2018, San Diego Marriott Mission Valley, San Diego, CA, USA.*, pages 315–323, 2018

4. Hanjun Dai* and Yingtao Tian* , Bo Dai, Steven Skiena, and Le Song. Syntax-directed variational autoencoder for structured data. In *International Conference on Learning Representations*, 2018

5. Muhao Chen, Yingtao Tian, Kai-Wei Chang, Steven Skiena, and Carlo Zaniolo. Co-training embeddings of knowledge graphs and entity descriptions for cross-lingual entity alignment. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018, July 13-19, 2018, Stockholm, Sweden.*, pages 3998–4004, 2018

6. Vivek Kulkarni, Yingtao Tian, Parth Dandiwala, and Steven Skiena. Simple neologism based domain independent models to predict year of authorship. In *Proceedings of the 27th International Conference on Computational Linguistics, COLING 2018, Santa Fe, New Mexico, USA, August 20-26, 2018*, pages 202–212, 2018

7. Mohammad Ruhul Amin, Alisa Yurovsky, Yingtao Tian, and Steven Skiena. Deepannotator: Genome annotation with deep learning. In *Proceedings of the 2018 ACM International Conference on Bioinformatics, Computational Biology, and Health Informatics*, BCB '18, pages 254–259, New York, NY, USA, 2018. ACM

8. Yingtao Tian, Haochen Chen, Bryan Perozzi, Muhao Chen, Xiaofei Sun, and Steven Skiena. Social relation inference via label propagation. In *Proceedings of the 41st European Conference on Information Retrieval, ECIR 2019, Cologe, Germany, April 14-18, 2018*, 2019

9. Haochen Chen, Xiaofei Sun, Yingtao Tian, Bryan Perozzi, Muhao Chen, and Steven Skiena. Enhanced network embeddings via exploiting edge labels. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management, CIKM 2018, Torino, Italy, October 22-26, 2018*, pages 1579–1582, 2018

10. Muhao Chen* and Yingtao Tian* , Haochen Chen, Kai-Wei Chang, Steven Skiena, and Carlo Zaniolo. Learning to represent bilingual dictionaries. In *Submitted*, 2019

11. Wenlu Wang, Yingtao Tian, Hongyu Xiong, Haixun Wang, and Wei-Shinn Ku. A transfer-learnable natural language interface for database. In *Submitted*, 2019

12. Yingtao Tian and Jesse Engel. Latent translation: Crossing modalities by bridging generative models. In *Submitted*, 2019

1 Introduction

Computationally modeling the real world starts with modeling the data that is observed from it. Data exists in a variety of modalities such as image like a photo of real-world objects, natural languages like online conversations, a formal language like line notation for chemical molecules, or complex relations such as ones in a knowledge base. By modeling these data, we open the door to leveraging rich real-world information for agents to behave more naturally in interaction with humans. However, it remains challenging to (a) how to modeling data from different modality and (b) how to model them in a unified way for downstream tasks to leverage.

The recent successes of representation learning that associates data with continuous vector representations pave direction for a unified way to model data. In this thesis, we address the challenge by studying the approaches in modeling data of multiple modalities using representation learning.

1.1 Two Whys

The combination of data in a variety of modalities and representation learning naturally leads to two questions for us to answer:

Why caring about data in a variety of modalities? Many data and application that are of interests of scientists and engineers are of different varieties with their own structures. One example could be modeling molecules with structured graphs where atoms are nodes and bonds are edges, which enables many tasks, such as modeling and drug search, with influential scientifically and industry application. Another example is computer programs, which presents important tasks such as code analysis, verification and security check. Failing to deal with them properly can leads to hundreds of millions of losses[1].

Why looking for representation learning for benefit? Representation, or sometimes called latent space, gives a representation, or a latent vector, z to data x, as shown in Figure 1.1.

[1]One good example is Ariane 5 rocket loss (https://hownot2code.com/2016/09/02/a-space-error-370-million-for-an-integer-overflow/).

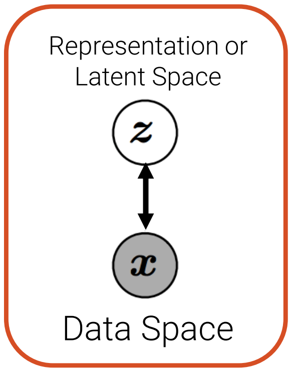


Figure 1.1: General diagram of representation learning.

The general philosophy behind my research that guides me to focus on representation learning is best represented by the following quotation from Bengio [14]: "a good representation ...captures the posterior distribution of the underlying explanatory factors for the observed input ...and also is useful as input to a supervised predictor. "

One famous example is word embedding which assigns a vector for each word in a natural language. It not only enables arithmetics in latent space such as "man - king + queen = woman", but can be used as features fed into downstream natural language processing tasks. Another benefit exists when we operate on latent space rather than data space when optimizing in data space is hard such as that for molecules case, searching for new molecules involves discrete searching consisting of many steps of adding, removing or substituting chemical functional groups. We could instead go to the representation, or the continuous latent space, and leverage continuous optimization such as Bayesian optimization here.

1.2 Thesis Overview

This thesis is organized as follows detailing my works on representation learning for different modalities of data under different circumstances. Data of multiple modalities are characterized by their respective structures that call for a variety of technique to deal with. One kind of data is structured ones, for which In Chapter 2, I present a generative model that captures the representations for discrete structures with formal grammars and semantics and generate both syntactically and semantically correct data. The other kind of data could be continuous, for which in Chapter 3, I propose a fast and effective label inference for social relations in collaboration networks which can be formalized as a multi-label classification problem on graph edges. The following Chapter 4 details my work on generating high-quality facial images of anime characters using generative adversarial network that are highly welcome by the community through the availability of web interface that runs on edge devices. Furthermore, I move forward with challenges concerning dealing simultaneously with data of more than one modality. Chapter 5 containing my work modeling bilingual dictionaries, or cross-lingual correspondence between sentences and lexicons, to benefit cross-lingual applications such as cross-lingual semantic search and question answering. In the end, I present in Chapter 6 a novel approach to bridging modularities by learning a post-hoc interface between two existing models to solve a new task and facilitate transferring across different modalities (e.g., image-to-audio) and even different types of generative models. These works are covered in short as follows:

Syntax-Directed Variational Autoencoder for Structured Data [61] Deep generative models have been enjoying success in modeling continuous data. However, it remains challenging to capture the representations for discrete structures with formal grammars and semantics, e.g., computer programs and molecular structures. How to generate both syntactically and semantically correct data still remains largely an open problem.

Inspired by the theory of compiler where syntax and semantics check is done via syntax-directed translation (SDT), we propose a novel syntax-directed variational autoencoder (SD-VAE) by introducing stochastic lazy attributes. This approach converts the offline SDT check into on-the-fly generated guidance for constraining the decoder. Comparing to the state-of-the-art methods, our approach enforces constraints on the output space so that the output will be not only syntactically valid but also semantically reasonable.

We evaluate the proposed model with a few applications in programming language and molecules, including reconstruction and program / molecule optimization. The results demonstrate the effectiveness in incorporating syntactic and semantic constraints in discrete generative models, which is significantly better than current state-of-the-art approaches.

Social Relation Inference via Label Propagation [146] Collaboration networks are a ubiquitous way to characterize the interactions between people. In this paper, we consider the problem of inferring social relations in collaboration networks, such as the fields that researchers collaborate in, or the categories of projects that Github users work on together.

Social relation inference can be formalized as a multi-label classification problem on graph edges, but many popular algorithms for semi-supervised learning on graphs only operate on the nodes of a graph. To bridge this gap, we propose a principled method which leverages the natural homophily present in collaboration networks. First, observing that the fields of collaboration for two people are usually at the intersection of their interests, we transform an edge labeling into node labels. Second, we use a label propagation algorithm to propagate node labels in the entire graph. Once the label distribution for all nodes has been obtained, we can easily infer the label distribution for all edges. Experiments on three large-scale collaboration networks demonstrate that our method outperforms the state-of-the-art methods for social relation inference by a large margin, in addition to running several orders of magnitude faster.

Towards the Automatic Anime Characters Creation with Generative Adversarial Networks [75] Automatic generation of facial images has been well studied after the Generative Adversarial Network(GAN) came out. There exist some attempts applying the GAN model to the problem of generating facial images of anime characters, but none of the existing work gives a promising result.

In this work, we explore the training of GAN models specialized on an anime facial image dataset. We address the issue from both the data and the model aspect, by collecting a more clean, well-suited dataset and leverage proper, empirical application of GAN. With quantitative analysis and case studies, we demonstrate that our efforts lead to a stable and high-quality model.

Moreover, to assist people with anime character design, we build a website with our pre-trained model available online, which makes the model easily accessible to the general public.

Learning to Represent Bilingual Dictionaries [115] Bilingual word embeddings have been widely used to capture the similarity of lexical semantics in different human languages. However, many applications, such as cross-lingual semantic search and question answering, can be largely benefited from the cross-lingual correspondence between sentences and lexicons.

To bridge this gap, we propose a neural embedding model that leverages bilingual dictionaries. The proposed model is trained to map the literal word definitions to the cross-lingual target words, for which we explore with different sentence encoding techniques. To enhance the learning process on limited resources, our model adopts several critical learning strategies, including multi-task learning on different bridges of languages, and joint learning of the dictionary model with a bilingual word embedding model.

Experimental evaluation focuses on two applications. The results of the cross-lingual reverse dictionary retrieval task show our model's promising ability of comprehending bilingual concepts based on descriptions, and highlight the effectiveness of proposed learning strategies in improving performance. Meanwhile, our model effectively addresses the bilingual paraphrase identification problem and significantly outperforms previous approaches.

Latent Translation: Crossing Modalities by Bridging Generative Models [147] End-to-end optimization has achieved state-of-the-art performance on many specific problems, but there is no straight-forward way to combine pretrained models for new problems.

Here, we explore improving modularity by learning a post-hoc interface between two existing models to solve a new task. Specifically, we take inspiration from neural machine translation, and cast the challenging problem of cross-modal domain transfer as unsupervised translation between the latent spaces of pretrained deep generative models. By abstracting away the data representation, we demonstrate that it is possible to transfer across different modalities (e.g., image-to-audio) and even different types of generative models (e.g., VAE-to-GAN). We compare to state-of-the-art techniques and find that a straight-forward variational autoencoder is able to best bridge the two generative models through learning a shared latent space.

We can further impose supervised alignment of attributes in both domains with a classifier in the shared latent space. Through qualitative and quantitative evaluations, we demonstrate that locality and semantic alignment are preserved through the transfer process, as indicated by high transfer accuracies and smooth interpolations within a class. Finally, we show this modular structure speeds up the training of new interface models by several orders of magnitude by decoupling it from expensive retraining of base generative models.

1.3 Extra Works

I have been fortunate to work on a broad spectrum of the works besides ones included in this thesis. Here I give a brief overview of them:

In the direction of modeling knowledge base using representation learning, I have contributed to several projects, including learning representation for multilingual knowledge base(or KB) that helps people in constructing a coherent knowledge base and assist machines in dealing with different expressions of entity relationships [30], devising representation learning mechanism for relation prediction in KB that tackles transitivity, symmetry and hierarchical relations that are common in semantic web community [29], and multilingual KB with views of relation and description using co-training and leverages a weakly aligned multilingual KG for semi-supervised cross-lingual learning using entity descriptions [28].

My work in natural language processing leads to a novel approach in general-purpose natural language interface for databases that bridges both complexity and expressiveness of natural languages and powerful database management systems can optimize and answer queries against any relational database [157]. Another work in this direction presents domain-independent models to date documents based only on neologism usage patterns, provides insights into the temporal locality of word usage and generalize to various domains [84].

I also applied my understanding of sequence modeling in deep learning to natural science, especially in Genome detection archive improved the performance by incorporating intermediate objectives and downstream algorithms to achieve better accuracy [5].

2 Syntax-Directed Variational Autoencoder for Structured Data [2]

2.1 Introduction

Recent advances in deep representation learning have resulted in powerful probabilistic generative models which have demonstrated their ability on modeling continuous data, *e.g.*, time series signals [151, 38] and images [122, 77]. Despite the success in these domains, it is still challenging to correctly generate discrete structured data, such as graphs, molecules and computer programs. Since many of the structures have syntax and semantic formalisms, the generative models without explicit constraints often produces invalid ones.

Conceptually an approach in generative model for structured data can be divided in two parts, one being the formalization of the structure generation and the other one being a (usually deep) generative model producing parameters for stochastic process in that formalization. Often the hope is that with the help of training samples and capacity of deep models, the loss function will prefer the valid patterns and encourage the mass of the distribution of the generative model towards the desired region automatically.

Arguably the simplest structured data are sequences, whose generation with deep model has been well studied under the seq2seq [139] framework that models the generation of sequence as a series of token choices parameterized by recurrent neural networks (RNNs). Its widespread success has encourage several pioneer works that consider the conversion of more complex structure data into sequences and apply sequence models to the represented sequences. [54] (CVAE) is a representative work of such paradigm for the chemical molecule generation, using the SMILES line notation [159] for representing molecules. However, because of the lack of formalization of syntax and semantics serving as the restriction of the *particular* structured data, underfitted *general-purpose* string generative models will often lead to invalid outputs. Therefore, to obtain a reasonable model via such training procedure, we need to prepare large amount of valid combinations of the

[2]The content of this section is taken from:
[61] Hanjun Dai* and Yingtao Tian* , Bo Dai, Steven Skiena, and Le Song. Syntax-directed variational autoencoder for structured data. In *International Conference on Learning Representations*, 2018

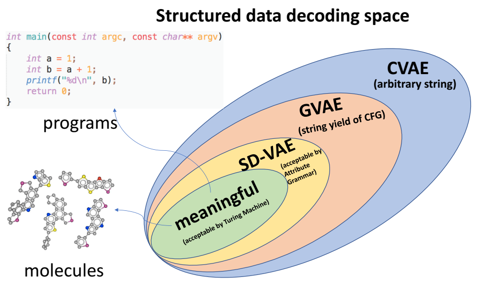
structures, which is time consuming or even not practical in domains like drug discovery.

To tackle such a challenge, one approach is to incorporate the structure restrictions explicitly into the generative model. For the considerations of computational cost and model generality, context-free grammars (CFG) have been taken into account in the decoder parametrization. For instance, in molecule generation tasks, [85] proposes a grammar variational autoencoder (GVAE) in which the CFG of SMILES notation is incorporated into the decoder. The model generates the parse trees directly in a top-down direction, by repeatedly expanding any nonterminal with its production rules. Although the CFG provides a mechanism for generating *syntactic valid* objects, it is still incapable to regularize the model for generating *semantic valid* objects [85]. For example, in molecule generation, the semantic of the SMILES languages requires that the rings generated must be closed; in program generation, the referenced variable should be defined in advance and each variable can only be defined exactly once in each local context (illustrated in Fig 2.1b). All the examples require cross-serial like dependencies which are not enforceable by CFG, implying that more constraints beyond CFG are needed to achieve semantic valid production in VAE.

In the theory of compiler, attribute grammars, or syntax-directed definition has been proposed for attaching semantics to a parse tree generated by context-free grammar. Thus one straightforward but not practical application of attribute grammars is, after generating a syntactic valid molecule candidate, to conduct offline semantic checking. This process needs to be repeated until a semantically valid one is discovered, which is at best computationally inefficient and at worst infeasible, due to extremely low rate of passing checking. As a remedy, we propose the *syntax-direct variational autoencoder* (SD-VAE), in which a semantic restriction component is advanced to the stage of syntax tree generator. This allows the generator with both syntactic and semantic validation. The proposed syntax-direct generative mechanism in the decoder further constraints the output space to ensure the semantic correctness in the tree generation process. The relationships between our proposed model and previous models can be characterized in Figure 2.1a.

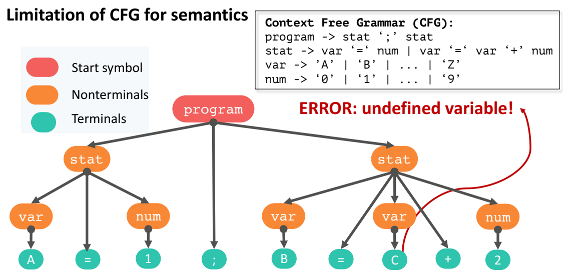
Our method brings theory of formal language into stochastic generative model. The contribution of our paper can be summarized as follows:

- *Syntax and semantics enforcement*: We propose a new formalization of


Structured data decoding space
programs
molecules
CVAE
(arbitrary string)
GVAE
(string yield of CFG)
SD-VAE
(acceptable by Attribute Grammar)
meaningful
(acceptable by Turing Machine)

```
int main(const int argc, const char** argv)
{
    int a = 1;
    int b = a + 1;
    printf("%d\n", b);
    return 0;
}
```

(a) Illustrations of structured data decoding space



```
Context Free Grammar (CFG):
program -> stat ‘;’ stat
stat -> var ‘=‘ num | var ‘=‘ var ‘+’ num
var -> ’A’ | ‘B’ | ... | ‘Z’
num -> ‘0’ | ‘1’ | ... | ‘9’
```

(b) CFG parses program ‘A=1;B=C+2’

Figure 2.1: Illustration on left shows the hierarchy of the structured data decoding space w.r.t different works and theoretical classification of corresponding strings from formal language theory. SD-VAE, our proposed model with attribute grammar reshapes the output space tighter to the meaningful target space than existing works. On the right we show a case where CFG is unable to capture the semantic constraints, since it successfully parses an invalid program.

semantics that systematically converts the offline semantic check into online guidance for stochastic generation using the proposed *stochastic lazy attribute*. This allows us effectively address both syntax and semantic constraints.

- *Efficient learning and inference*: Our approach has computational cost $O(n)$ where n is the length of structured data. This is the same as existing methods like CVAE and GVAE which do not enforce semantics in generation. During inference, the SD-VAE runs with semantic guiding on-the-fly, while the existing alternatives generate many candidates for semantic checking.

- *Strong empirical performance*: We demonstrate the effectiveness of the SD-VAE through applications in two domains, namely (1) the subset of Python programs and (2) molecules. Our approach consistently and significantly improves the results in evaluations including generation, reconstruction and optimization.

2.2 Background

Before introducing our model and the learning algorithm, we first provide some background knowledge which is important for understanding the proposed method.

2.2.1 Variational Autoencoder

The variational autoencoder [79, 126] provides a framework for learning the probabilistic generative model as well as its posterior, respectively known as decoder and encoder. We denote the observation as x, which is the structured data in our case, and the latent variable as z. The decoder is modeling the probabilistic generative processes of x given the continuous representation z through the likelihood $p_\theta(x|z)$ and the prior over the latent variables $p(z)$, where θ denotes the parameters. The encoder approximates the posterior $p_\theta(z|x) \propto p_\theta(x|z)p(z)$ with a model $q_\psi(z|x)$ parametrized by ψ. The decoder and encoder are learned simultaneously by maximizing the evidence lower bound (ELBO) of the marginal likelihood, *i.e.*,

$$\begin{aligned}\mathcal{L}\left(X;\theta,\psi\right) &:= \sum_{x\in X} \mathbb{E}_{q(z|x)}\left[\log p_\theta(x|z)p(z) - \log q_\psi(z|x)\right] \\ &\leq \sum_{x\in X} \log \int p_\theta(x|z)p(z)dz,\end{aligned} \tag{1}$$

where X denotes the training datasets containing the observations.

2.2.2 Context Free Grammar and Attribute Grammar

Context free grammar A context free grammar (CFG) is defined as $G = \langle \mathcal{V}, \Sigma, \mathcal{R}, s\rangle$, where symbols are divided into $\mathcal{V}$, the set of non-terminal symbols, Σ, the set of terminal symbols and $s \in \mathcal{V}$, the start symbol. Here $\mathcal{R}$ is the set of production rules. Each production rule $r \in \mathcal{R}$ is denoted as $r = \alpha \to \beta$ for $\alpha \in \mathcal{V}$ is a nonterminal symbol, and $\beta = u_1u_2\ldots u_{|\beta|} \in (\mathcal{V}\bigcup\Sigma)^*$ is a sequence of terminal and/or nonterminal symbols.

Attribute grammar To enrich the CFG with "semantic meaning", [81] formalizes attribute grammar that introduces attributes and rules to CFG. An attribute is an attachment to the corresponding nonterminal symbol in CFG, written in the format $\langle v\rangle.a$ for $v \in \mathcal{V}$. There can be two types of attributes assigned to non-terminals in G: the *inherited* attributes and the *synthesized* attributes. An inherited attribute depends on the attributes from its parent and siblings, while a synthesized attribute is computed based on the attributes of its children. Formally, for a production $u_0 \rightarrow u_1 u_2 \ldots u_{|\beta|}$, we denote $I(u_i)$ and $S(u_i)$ be the sets of *inherited* and *synthesized* attributes of u_i for $i \in \{0, \ldots, |\beta|\}$, respectively.

A motivational example We here exemplify how the above defined attribute grammar enriches CFG with non-context-free semantics. We use the following toy grammar, a subset of SMILES that generates either a chain or a cycle with three carbons:

Production	**Semantic Rule**
$\langle s \rangle \rightarrow \langle atom \rangle_1$ 'C' $\langle atom \rangle_2$	$\langle s \rangle.\texttt{matched} \leftarrow \langle atom \rangle_1.\texttt{set} \bigcap \langle atom \rangle_2.\texttt{set}$, $\langle s \rangle.\texttt{ok} \leftarrow \langle atom \rangle_1.\texttt{set} = \langle s \rangle.\texttt{matched} = \langle atom \rangle_2.\texttt{set}$
$\langle atom \rangle \rightarrow$ 'C' | 'C' $\langle bond \rangle$ $\langle digit \rangle$	$\langle atom \rangle.\texttt{set} \leftarrow \varnothing \mid \mathrm{concat}(\langle bond \rangle.\texttt{val}, \langle digit \rangle.\texttt{val})$
$\langle bond \rangle \rightarrow$ '-' | '=' | '#'	$\langle bond \rangle.\texttt{val} \leftarrow$ '-' | '=' | '#'
$\langle digit \rangle \rightarrow$ '1' | '2' | ... | '9'	$\langle digit \rangle.\texttt{val} \leftarrow$ '1' | '2' ... | '9'

where we show the production rules in CFG with $\rightarrow$ on the left, and the calculation of attributes in attribute grammar with $\leftarrow$ on the left. Here we leverage the attribute grammar to check (with attribute `matched`) whether the ringbonds come in pairs: a ringbond generated at $\langle atom \rangle_1$ should match the bond type and bond index that generated at $\langle atom \rangle_2$, also the semantic constraint expressed by $\langle s \rangle$`.ok` requires that there is no difference between the `set` attribute of $\langle atom \rangle_1$ and $\langle atom \rangle_2$. Such constraint in SMILES is known as *cross-serial dependencies* (CSD) [22] which is non-context-free [136]. See Appendix 2.A.3 for more explanations. Figure 2.2 illustrates the process of performing syntax and semantics check in compilers. Here all the attributes are *synthetic*, *i.e.*, calculated in a bottom-up direction.

So generally, in the semantic correctness checking procedure, one need to perform bottom-up procedures for calculating the attributes *after* the parse tree is generated. However, in the top-down structure generating process, the parse tree is not ready for semantic checking, since the synthesized attributes of each node require information from its children nodes, which are not generated yet. Due to such dilemma, it is nontrivial to use the attribute grammar to guide the top-down generation of the tree-structured data. One straightforward way is using acceptance-rejection sampling scheme, *i.e.*, using the decoder of CVAE or GVAE as a proposal and the semantic checking as the threshold. It is obvious that since the decoder does not include semantic

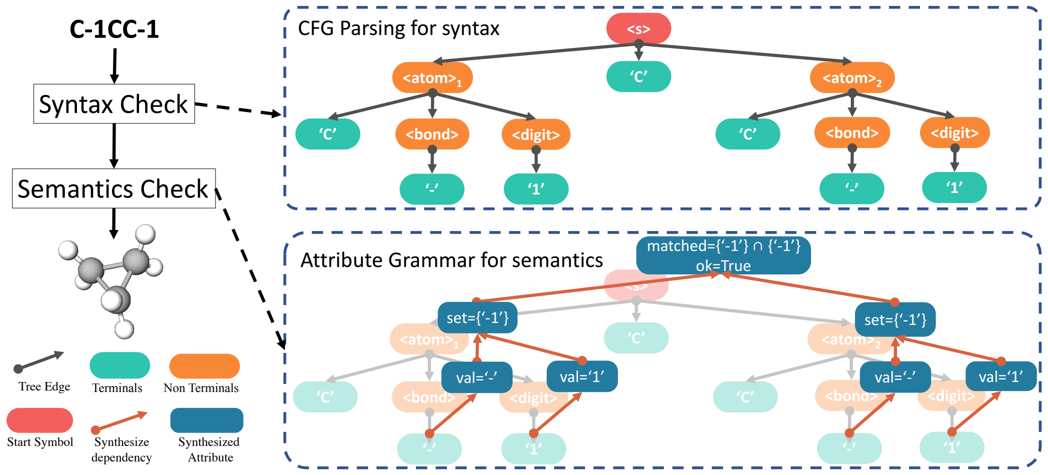


Figure 2.2: Bottom-up syntax and semantics check in compilers.

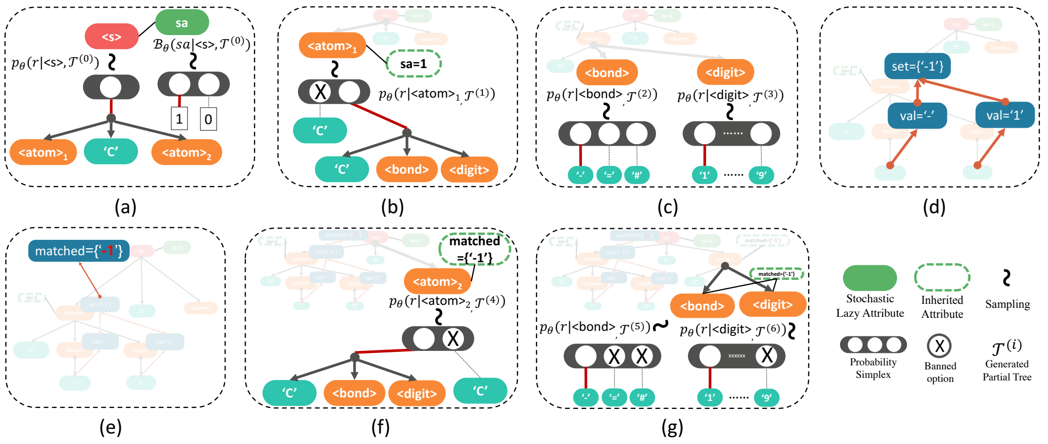
guidance, the proposal distribution may raise semantically invalid candidate frequently, therefore, wasting the computational cost in vain.

2.3 Syntax-Directed Variational Autoencoder

As described in Section 2.2.2, directly using attribute grammar in an offline fashion (*i.e.*, after the generation process finishes) is not efficient to address both syntax and semantics constraints. In this section we describe how to bring forward the attribute grammar online and incorporate it into VAE, such that our VAE addresses both *syntactic* and *semantic* constraints. We name our proposed method Syntax-Directed Variational Autoencoder (SD-VAE).

2.3.1 Stochastic Syntax-Directed Decoder

By scrutinizing the tree generation, the major difficulty in incorporating the attributes grammar into the processes is the appearance of the synthesized attributes. For instance, when expanding the start symbol $\langle s \rangle$, none of its children is generated yet. Thus their attributes are also absent at this time, making the $\langle s \rangle$.`matched` unable to be computed. To enable the on-the-fly computation of the synthesized attributes for semantic validation during tree generation, besides the two types of attributes, we introduce the *stochastic lazy attributes* to enlarge the existing attribute grammar. Such *stochasticity* transforms the corresponding synthesized attribute into inherited constraints



<s>
sa
$p_\theta(r|\text{<s>}, \mathcal{T}^{(0)})$
$\mathcal{B}_\theta(sa|\text{<s>}, \mathcal{T}^{(0)})$
1
0
<atom>₁
'C'
<atom>₂
(a)
sa=1
$p_\theta(r|\text{<atom>}_1, \mathcal{T}^{(1)})$
<bond>
<digit>
(b)
$p_\theta(r|\text{<bond>}, \mathcal{T}^{(2)})$
$p_\theta(r|\text{<digit>}, \mathcal{T}^{(3)})$
'-'
'='
'#'
'1'
'9'
(c)
set={'-1'}
val='-'
val='1'
(d)
matched={'-1'}
(e)
matched ={'-1'}
$p_\theta(r|\text{<atom>}_2, \mathcal{T}^{(4)})$
(f)
$p_\theta(r|\text{<bond>}, \mathcal{T}^{(5)})$
$p_\theta(r|\text{<digit>}, \mathcal{T}^{(6)})$
(g)
Stochastic Lazy Attribute
Inherited Attribute
Sampling
Probability Simplex
Banned option
$\mathcal{T}^{(i)}$ Generated Partial Tree

Figure 2.3: On-the-fly generative process of SD-VAE in order from (a) to (g). Steps: (a) stochastic generation of attribute; (b)(f)(g) constrained sampling with inherited attributes; (c) unconstrained sampling; (d) synthesized attribute calculation on generated subtree. (e) lazy evaluation of the attribute at root node.

in generative procedure; and lazy linking mechanism sets the actual value of the attribute, once all the other dependent attributes are ready. We demonstrate how the decoder with *stochastic lazy attributes* will generate semantic valid output through the same pedagogical example as in Section 2.2.2. Figure 2.3 visually demonstrates this process.

The tree generation procedure is indeed sampling from the decoder $p_\theta(x|z)$, which can be decomposed into several steps that elaborated below:

i) stochastic predetermination: in Figure 2.3(a), we start from the node $\langle s \rangle$ with the synthesized attributes $\langle s \rangle$.`matched` determining the index and bond type of the ringbond that will be matched at node $\langle s \rangle$. Since we know nothing about the children nodes right now, the only thing we can do is to 'guess' a value. That is to say, we associate a stochastic attribute $\langle s \rangle$.`sa` $\in \{0,1\}^{C_a} \sim \prod_{i=1}^{C_a} \mathcal{B}(sa_i|z)$ as a predetermination for the sake of the absence of synthesized attribute $\langle s \rangle$.`matched`, where $\mathcal{B}(\cdot)$ is the Bernoulli distribution. Here C_a is the maximum cardinality possible [3] for the corresponding attribute a. In above example, the 0 indicates no ringbond and 1 indicates one ringbond at both $\langle atom \rangle_1$ and $\langle atom \rangle_2$, respectively.

[3]Note that setting threshold for C_a assumes a *mildly context sensitive grammar* (*e.g.*, limited CSD).

Algorithm 1 Decoding with Stochastic Syntax-Directed Decoder

```
1: Global variables: CFG: $G = (\mathcal{V}, \Sigma, \mathcal{R}, s)$, decoder network parameters $\theta$
2: procedure GenTree($node$, $\mathcal{T}$)
3:     Sample stochastic lazy attribute $node._{sa} \sim \mathcal{B}_\theta(sa|node, \mathcal{T})$    ▷ when introduced on $node$
4:     Sample production rule $r = (\alpha \to \beta) \in \mathcal{R} \sim p_\theta(r|ctx, node, \mathcal{T})$.    ▷ The conditioned variables encodes the semantic constraints in tree generation.
5:     $ctx \leftarrow \text{RNN}(ctx, r)$    ▷ update context vector
6:     for $i = 1, \ldots, |\beta|$ do
7:         $v_i \leftarrow \text{Node}(u_i, node, \{v_j\}_{j=1}^{i-1})$    ▷ node creation with parent and siblings' attributes
8:         GenTree($v_i$, $\mathcal{T}$)    ▷ recursive generation of children nodes
9:         Update synthetic and stochastic attributes of $node$ with $v_i$    ▷ Lazy linking
10:    end for
11: end procedure
```

ii) constraints as inherited attributes: we pass the $\langle s \rangle$.`sa` as inherited constraints to the children of node $\langle s \rangle$, *i.e.*, $\langle atom \rangle_1$ and $\langle atom \rangle_2$ to ensure the semantic validation in the tree generation. For example, Figure 2.3(b) `'sa=1'` is passed down to $\langle atom \rangle_1$.

iii) sampling under constraints: without loss of generality, we assume $\langle atom \rangle_1$ is generated before $\langle atom \rangle_2$. We then sample the rules from $p_\theta(r|\langle atom \rangle_1, \langle s \rangle, z)$ for expanding $\langle atom \rangle_1$, and so on and so forth to generate the subtree recursively. Since we carefully designed sampling distribution that is conditioning on the stochastic property, the inherited constraints will be eventually satisfied. In the example, due to the $\langle s \rangle$.`sa` = `'1'`, when expanding $\langle atom \rangle_1$, the sampling distribution $p_\theta(r|\langle atom \rangle_1, \langle s \rangle, z)$ only has positive mass on rule $\langle atom \rangle \to$ `'C'` $\langle bond \rangle$ $\langle digit \rangle$.

iv) lazy linking: once we complete the generation of the subtree rooted at $\langle atom \rangle_1$, the synthesized attribute $\langle atom \rangle_1$.`set` is now available. According to the semantic rule for $\langle s \rangle$.`matched`, we can instantiate $\langle s \rangle$.`matched` = $\langle atom \rangle_1$.`set` = {`'-1'`}. This linking is shown in Figure 2.3(d)(e). When expanding $\langle atom \rangle_2$, the $\langle s \rangle$.`matched` will be passed down as inherited attribute to regulate the generation of $\langle atom \rangle_2$, as is demonstrated in Figure 2.3(f)(g).

In summary, the general syntax tree $\mathcal{T} \in L(G)$ can be constructed step by step, within the languages $L(G)$ covered by grammar G. In the beginning,

$\mathcal{T}^{(0)} = root$, where $root._{symbol} = s$ which contains only the start symbol s. At step t, we will choose an nonterminal node in the *frontier*[4] of partially generated tree $\mathcal{T}^{(t)}$ to expand. The generative process in each step $t = 0, 1, \ldots$ can be described as:

1. Pick node $v^{(t)} \in Fr(\mathcal{T}^{(t)})$ where its attributes needed are either satisfied, or are stochastic attributes that should be sampled first according to Bernoulli distribution $\mathcal{B}(\cdot|v^{(t)}, \mathcal{T}^{(t)})$;
2. Sample rule $r^{(t)} = \alpha^{(t)} \to \beta^{(t)} \in \mathcal{R}$ according to distribution $p_\theta(r^{(t)}|v^{(t)}, \mathcal{T}^{(t)})$, where $v^{(t)}._{symbol} = \alpha^{(t)}$, and $\beta^{(t)} = u_1^{(t)} u_2^{(t)} \ldots u_{|\beta^{(t)}|}^{(t)}$, *i.e.*, expand the nonterminal with production rules defined in CFG.
3. $\mathcal{T}^{(t+1)} = \mathcal{T}^{(t)} \bigcup \{(v^{(t)}, u_i^{(t)})\}_{i=1}^{|\beta^{(t)}|}$, *i.e.*, grow the tree by attaching $\beta^{(t)}$ to $v^{(t)}$. Now the node $v^{(t)}$ has children represented by symbols in $\beta^{(t)}$.

The above process continues until all the nodes in the frontier of $\mathcal{T}^{(T)}$ are all terminals after T steps. Then, we obtain the algorithm 1 for sampling both syntactic and semantic valid structures.

In fact, in the model training phase, we need to compute the likelihood $p_\theta(x|z)$ given x and z. The probability computation procedure is similar to the sampling procedure in the sense that both of them requires tree generation. The only difference is that in the likelihood computation procedure, the tree structure, *i.e.*, the computing path, is fixed since x is given; While in the sampling procedure, it is sampled following the learned model. Specifically, the generative likelihood can be written as:

$$p_\theta(x|z) = \prod_{t=0}^{T} p_\theta(r_t|ctx^{(t)}, node^{(t)}, \mathcal{T}^{(t)}) \mathcal{B}_\theta(sa_t|node^{(t)}, \mathcal{T}^{(t)}) \tag{2}$$

where $ctx^{(0)} = z$ and $ctx^{(t)} = \text{RNN}(r_t, ctx^{(t-1)})$. Here RNN can be commonly used LSTM, *etc.*.

2.3.2 Structure-Based Encoder

As we introduced in section 2.2, the encoder, $q_\psi(z|x)$ approximates the posterior of the latent variable through the model with some parametrized function with parameters ψ. Since the structure in the observation x plays an important role, the encoder parametrization should take care of such information.

[4]Here frontier is the set of all nonterminal leaves in current tree.

The recently developed deep learning models [45, 37, 93] provide powerful candidates as encoder. However, to demonstrate the benefits of the proposed syntax-directed decoder in incorporating the attribute grammar for semantic restrictions, we will exploit the same encoder in [85] for a fair comparison later.

We provide a brief introduction to the particular encoder model used in [85] for a self-contained purpose. Given a program or a SMILES sequence, we obtain the corresponding parse tree using CFG and decompose it into a sequence of productions through a pre-order traversal on the tree. Then, we convert these productions into one-hot indicator vectors, in which each dimension corresponds to one production in the grammar. We will use a deep convolutional neural networks which maps this sequence of one-hot vectors to a continuous vector as the encoder.

2.3.3 Model Learning

Our learning goal is to maximize the evidence lower bound in Eq 1. Given the encoder, we can then map the structure input into latent space z. The variational posterior $q(z|x)$ is parameterized with Gaussian distribution, where the mean and variance are the output of corresponding neural networks. The prior of latent variable $p(z) = \mathcal{N}(0, I)$. Since both the prior and posterior are Gaussian, we use the closed form of KL-divergence that was proposed in [79]. In the decoding stage, our goal is to maximize $p_\theta(x|z)$. Using the Equation (2), we can compute the corresponding conditional likelihood. During training, the syntax and semantics constraints required in Algorithm 1 can be precomputed. In practice, we observe no significant time penalty measured in wall clock time compared to previous works.

2.4 Related work

Generative models with discrete structured data have raised increasing interests among researchers in different domains. The classical sequence to sequence model [139] and its variations have also been applied to molecules [54]. Since the model is quite flexible, it is hard to generate valid structures with limited data, though [70] shows that an extra validator model could be helpful to some degree. Techniques including data augmentation [18], active learning [71] and reinforcement learning [58] also been proposed to tackle this issue. However, according to the empirical evaluations from [15], the validity is still not satisfactory. Even when the validity is enforced, the models tend to overfit to simple structures while neglect the diversity.

Since the structured data often comes with formal grammars, it is very helpful to generate its parse tree derived from CFG, instead of generating sequence of tokens directly. The Grammar VAE[85] introduced the CFG constrained decoder for simple math expression and SMILES string generation. The rules are used to mask out invalid syntax such that the generated sequence is always from the language defined by its CFG. [119] uses a RecursiveReverse-Recursive Neural Network (R3NN) to capture global context information while expanding with CFG production rules. Although these works follow the syntax via CFG, the context sensitive information can only be captured using variants of sequence/tree RNNs [3, 44, 170], which may not be time and sample efficient.

In our work, we capture the semantics with proposed stochastic lazy attributes when generating structured outputs. By addressing the most common semantics to harness the deep networks, it can greatly reshape the output domain of decoder [66]. As a result, we can also get a better generative model for discrete structures.

2.5 Experiments

We show the effectiveness of our proposed SD-VAE with applications in two domains, namely programs and molecules. We compare our method with CVAE [54] and GVAE [85]. CVAE only takes character sequence information, while GVAE utilizes the context-free grammar. To make a fair comparison, we closely follow the experimental protocols that were set up in [85]. The training details are included in Appendix 2.B.

Our method gets significantly better results than previous works. It yields better reconstruction accuracy and prior validity by large margins, while also having comparative diversity of generated structures. More importantly, the SD-VAE finds better solution in program and molecule regression and optimization tasks. This demonstrates that the continuous latent space obtained by SD-VAE is also smoother and more discriminative.

2.5.1 Settings

Here we first describe our datasets in detail. The programs are represented as a list of statements. Each statement is an atomic arithmetic operation on variables (labeled as `v0`, `v1`, $\cdots$, `v9`) and/or immediate numbers $(1, 2, \ldots, 9)$. Some examples are listed below:

```
 v3=sin(v0);v8=exp(2);v9=v3-v8;v5=v0*v9;return:v5
v2=exp(v0);v7=v2*v0;v9=cos(v7);v8=cos(v9);return:v8
```

Here `v0` is always the input, and the variable specified by `return` (respectively `v5` and `v8` in the examples) is the output, therefore it actually represent univariate functions $f : \mathbb{R} \to \mathbb{R}$. Note that a correct program should, besides the context-free grammar specified in Appendix 2.A.1, also respect the semantic constraints. For example, a variable should be defined before being referenced. We randomly generate $130,000$ programs, where each consisting of 1 to 5 valid statements. Here the maximum number of decoding steps $T = 80$. We hold 2000 programs out for testing and the rest for training and validation.

For molecule experiments, we use the same dataset as in [85]. It contains $250,000$ SMILES strings, which are extracted from the ZINC database [54]. We use the same split as [85], where 5000 SMILES strings are held out for testing. Regarding the syntax constraints, we use the grammar specified in Appendix 2.A.2, which is also the same as [85]. Here the maximum number of decoding steps $T = 278$.

For our SD-VAE, we address some of the most common semantics:

Program semantics We address the following: *a*) variables should be defined before use, *b*) program must return a variable, *c*) number of statements should be less than 10.

Molecule semantics The SMILES semantics we addressed includes: *a*) ringbonds should satisfy cross-serial dependencies, *b*) explicit valence of atoms should not go beyond permitted. For more details about the semantics of SMILES language, please refer to Appendix 2.A.3.

2.5.2 Reconstruction Accuracy and Prior Validity

We use the held-out dataset to measure the reconstruction accuracy of VAEs. For prior validity, we first sample the latent representations from prior distribution, and then evaluate how often the model can decode into a valid structure. Since both encoding and decoding are stochastic in VAEs, we follow the Monte Carlo method used in [85] to do estimation:

a) *reconstruction:* for each of the structured data in the held-out dataset, we encode it 10 times and decoded (for each encoded latent space representation) 25 times, and report the portion of decoded structures that are the

	Program		Zinc SMILES	
Methods	**Reconstruction %***	**Valid Prior %**	**Reconstruction %**	**Valid Prior %**
SD-VAE	**96.46** (**99.90**, **99.12**, **90.37**)	**100.00**	**76.2**	**43.5**
GVAE	71.83 (96.30, 77.28, 41.90)	2.96	53.7	7.2
CVAE	13.79 (40.46, 0.87, 0.02)	0.02	44.6	0.7

Table 2.1: Reconstructing Accuracy and Prior Validity estimated using Monte Carlo method. Our proposed method (SD-VAE) performance significantly better than existing works.
* We also report the reconstruction % grouped by number of statements (3, 4, 5) in parentheses.

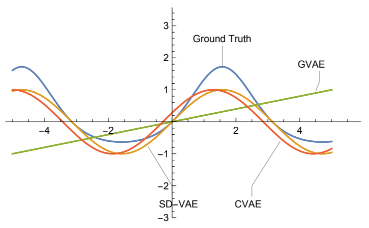
same as the input ones; *b*) *validity of prior:* we sample 1000 latent representations $\mathbf{z} \sim \mathcal{N}(O, \mathbf{I})$. For each of them we decode 100 times, and calculate the portion of 100,000 decoded results that corresponds to valid Program or SMILES sequences.

Program We show in the left part of Table 2.1 that our model has near perfect reconstruction rate, and most importantly, a perfect valid decoding program from prior. This huge improvement is due to our model that utilizes the full semantics that previous work ignores, thus in theory guarantees perfect valid prior and in practice enables high reconstruction success rate. For a fair comparison, we run and tune the baselines in 10% of training data and report the best result. In the same place we also report the reconstruction successful rate grouped by number of statements. It is shown that our model keeps high rate even with the size of program growing.

SMILES Since the settings are exactly the same, we include CVAE and GVAE results directly from [85]. We show in the right part of Table 2.1 that our model produces a much higher rate of successful reconstruction and ratio of valid prior. Figure 2.8 in Appendix 2.C.2 also demonstrates some decoded molecules from our method. Note that the results we reported have not included the semantics specific to aromaticity into account. If we use an alternative kekulized form of SMILES to train the model, then the valid portion of prior can go up to 97.3%.

2.5.3 Bayesian Optimization

One important application of VAEs is to enable the optimization (*e.g.*, find new structures with better properties) of discrete structures in continuous latent space, and then use decoder to obtain the actual structures. Following the protocol used in [85], we use Bayesian Optimization (BO) to search the programs and molecules with desired properties in latent space. Details about BO settings and parameters can be found in Appendix 2.C.1.



Method	Program	Score
CVAE	`v7=5+v0;v5=cos(v7);return:v5`	0.1742
	`v2=1-v0;v9=cos(v2);return:v9`	0.2889
	`v5=4+v0;v3=cos(v5);return:v3`	0.3043
GVAE	`v3=1/5;v9=-1;v1=v0*v3;return:v3`	0.5454
	`v2=1/5;v9=-1;v7=v2+v2;return:v7`	0.5497
	`v2=1/5;v5=-v2;v9=v5*v5;return:v9`	0.5749
SD-VAE	`v6=sin(v0);v5=exp(3);v4=v0*v6;return:v6`	**0.1206**
	`v5=6+v0;v6=sin(v5);return:v6`	**0.1436**
	`v6=sin(v0);v4=sin(v6);v5=cos(v4);v9=2/v4;return:v4`	**0.1456**
Ground Truth	`v1=sin(v0);v2=exp(v1);v3=v2-1;return:v3`	—

Figure 2.4: On the left are best programs found by each method using Bayesian Optimization. On the right are top 3 closest programs found by each method along with the distance to ground truth (lower distance is better). Both our SD-VAE and CVAE can find similar curves, but our method aligns better with the ground truth. In contrast the GVAE fails this task by reporting trivial programs representing linear functions.

Finding program In this application the models are asked to find the program which is most similar to the ground truth program. Here the distance is measured by $\log(1 + \text{MSE})$, where the MSE (Mean Square Error) calculates the discrepancy of program outputs, given the 1000 different inputs `v0` sampled evenly in $[-5, 5]$. In Figure 2.4 we show that our method finds the best program to the ground truth one compared to CVAE and GVAE.

Molecules Here we optimize the drug properties of molecules. In this problem, we ask the model to optimize for octanol-water partition coefficients (a.k.a *log P*), an important measurement of drug-likeness of a given molecule. As [54] suggests, for drug-likeness assessment *log P* is penalized by other properties including synthetic accessibility score [50]. In Figure 2.5 we show the the top-3 best molecules found by each method, where our method found molecules with better scores than previous works. Also one can see the molecule structures found by SD-VAE are richer than baselines, where the latter ones mostly consist of chain structure.

2.5.4 Predictive Performance of Latent Representation

The VAEs also provide a way to do unsupervised feature representation learning [54]. In this section, we seek to to know how well our latent space predicts the properties of programs and molecules. After the training of VAEs, we

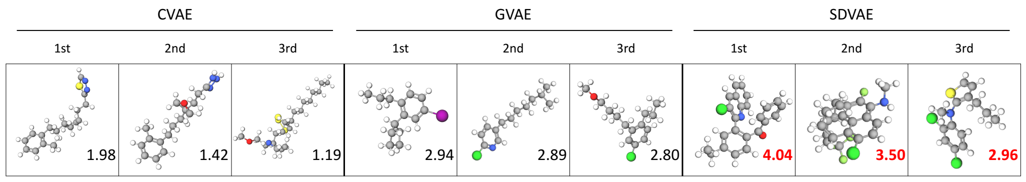


Figure 2.5: Best top-3 molecules and the corresponding scores found by each method using Bayesian Optimization.

	Program		**Zinc**	
Method	**LL**	**RMSE**	**LL**	**RMSE**
CVAE	-4.943 ± 0.058	3.757 ± 0.026	-1.812 ± 0.004	1.504 ± 0.006
GVAE	-4.140 ± 0.038	3.378 ± 0.020	-1.739 ± 0.004	1.404 ± 0.006
SD-VAE	**-3.754 ± 0.045**	**3.185 ± 0.025**	**-1.697 ± 0.015**	**1.366 ± 0.023**

Table 2.2: Predictive performance using encoded mean latent vector. Test LL and RMSE are reported.

dump the latent vectors of each structured data, and train the sparse Gaussian Process with the target value (namely the error for programs and the drug-likeness for molecules) for regression. We test the performance in the held-out test dataset. In Table 2.2, we report the result in Log Likelihood (LL) and Regression Mean Square Error (RMSE), which show that our SD-VAE always produces latent space that are more discriminative than both CVAE and GVAE baselines. This also shows that, with a properly designed decoder, the quality of encoder will also be improved via end-to-end training.

2.5.5 Diversity of Generated Molecules

Inspired by [15], here we measure the diversity of generated molecules as an assessment of the methods. The intuition is that a good generative model should be able to generate diverse data and avoid mode collapse in the learned space. We conduct this experiment in the SMILES dataset. We first sample 100 points from the prior distribution. For each point, we associate it with a molecule, which is the most frequent occurring valid SMILES decoded (we use 50 decoding attempts since the decoding is stochastic). We then, with one of the several molecular similarity metrics, compute the pairwise similarity and report the mean and standard deviation in Table 2.3. We see both methods do not have the mode collapse problem, while producing similar diversity

Similarity Metric	MorganFp	MACCS	PairFp	TopologicalFp
GVAE	**0.92 ± 0.10**	**0.83 ± 0.15**	0.94 ± 0.10	0.71 ± 0.14
SD-VAE	**0.92 ± 0.09**	**0.83 ± 0.13**	**0.95 ± 0.08**	**0.75 ± 0.14**

Table 2.3: Diversity as statistics from pair-wise distances measured as $1 - s$, where s is one of the similarity metrics. So higher values indicate better diversity. We show mean $\pm$ stddev of $\binom{100}{2}$ pairs among 100 molecules. Note that we report results from GVAE and our SD-VAE, because CVAE has very low valid priors, thus completely only failing this evaluation protocol.

scores. It indicates that although our method has more restricted decoding space than baselines, the diversity is not sacrificed. This is because we never rule-out the valid molecules. And a more compact decoding space leads to much higher probability in obtaining valid molecules.

2.5.6 Visualizing the Latent Space

We seek to visualize the latent space as an assessment of how well our generative model is able to produces a coherent and smooth space of program and molecules.

Program Following [21], we visualize the latent space of program by interpolation between two programs. More specifically, given two programs which are encoded to p_a and p_b respectively in the latent space, we pick 9 evenly interpolated points between them. For each point, we pick the corresponding most decoded structure. In Table 2.4 we compare our results with previous works. Our SD-VAE can pass though points in the latent space that can be decoded into valid programs without error and with visually more smooth interpolation than previous works. Meanwhile, CVAE makes both syntactic and semantic errors, and GVAE produces only semantic errors (reference of undefined variables), but still in a considerable amount.

SMILES For molecules, we visualize the latent space in 2 dimensions. We first embed a random molecule from the dataset into latent space. Then we randomly generate 2 orthogonal unit vectors A. To get the latent representation of neighborhood, we interpolate the 2-D grid and project back to latent space with pseudo inverse of A. Finally we show decoded molecules. In Figure 2.6, we present two of such grid visualizations. Subjectively compared with figures in [85], our visualization is characterized by having smooth differences between neighboring molecules, and more complicated decoded

CVAE	GVAE	SD-VAE
v6=cos(7);v8=exp(9);v2=v8⋆v0;v9=v2/v6;return:v9	v6=cos(7);v8=exp(9);v2=v8⋆v0;v9=v2/v6;return:v9	v6=cos(7);v8=exp(9);v2=v8⋆v0;v9=v2/v6;return:v9
v8=cos(3);v7=exp(7);v5=v7⋆v0;v9=v9/v6;return:v9	v3=cos(8);v6=exp(9);v6=v8⋆v0;v9=v2/v6;return:v9	v6=cos(7);v8=exp(9);v2=v8⋆v0;v9=v2/v6;return:v9
v4=cos(3);v8=exp(3);v2=v2⋆v0;v9=v8/v6;return:v9	v3=cos(8);v6=2/8;v6=v5⋆v9;v5=v8v5;return:v5	v6=cos(7);v8=exp(9);v3=v8⋆v0;v9=v3/v8;return:v9
v6=cos(3);v8=sin(3);v5=v4⋆1;v5=v3/v4;return:v9	v3=cos(6);v6=2/9;v6=v5+v5;v5=v1+v6;return:v5	v6=cos(7);v8=v6/9;v1=7⋆v0;v7=v6/v1;return:v7
v9=cos(1);v7=sin(1);v3=v1⋆5;v9=v9+v4;return:v9	v5=cos(6);v1=2/9;v6=v3+v2;v2=v5-v6;return:v2	v6=cos(7);v8=v6/9;v1=7⋆v6;v7=v6+v1;return:v7
v6=cos(1);v3=sin(10;;v9=8⋆v8;v7=v2/v2;return:v9	v5=sin(5);v3=v1/9;v6=v3-v3;v2=v7-v6;return:v2	v6=cos(7);v8=v6/9;v1=7⋆v8;v7=v6+v8;return:v7
v5=exp(v0;v4=sin(v0);v3=8⋆v1;v7=v3/v2;return:v9	v1=sin(1);v5=v5/2;v6=v2-v5;v2=v0-v6;return:v2	v6=exp(v0);v8=v6/2;v9=6⋆v8;v7=v9+v9;return:v7
v5=exp(v0);v1=sin(1);v5=2⋆v3;v7=v3+v8;return:v7	v1=sin(1);v7=v8/2;v8=v7/v9;v4=v4-v8;return:v4	v6=exp(v0);v8=v6-4;v9=6⋆v8;v7=v9+v8;return:v7
v4=exp(v0);v1=v7-8;v9=8⋆v3;v7=v3+v8;return:v7	v8=sin(1);v2=v8/2;v8=v0/v9;v4=v4-v8;return:v4	v6=exp(v0);v8=v6-4;v9=6⋆v6;v7=v9+v8;return:v7
v4=exp(v0);v9=v6-8;v6=2⋆v5;v7=v3+v8;return:v7	v6=exp(v0);v2=v6-4;v8=v0⋆v1;v7=v4+v8;return:v7	v6=exp(v0);v8=v6-4;v4=4⋆v6;v7=v4+v8;return:v7
v6=exp(v0);v8=v6-4;v4=4⋆v8;v7=v4+v8;return:v7	v6=exp(v0);v8=v6-4;v4=4⋆v8;v7=v4+v8;return:v7	v6=exp(v0);v8=v6-4;v4=4⋆v8;v7=v4+v8;return:v7

Table 2.4: Interpolation between two valid programs (the top and bottom ones in brown) where each program occupies a row. Programs in red are with syntax errors. Statements in blue are with semantic errors such as referring to unknown variables. Rows without coloring are correct programs. Observe that when a model passes points in its latent space, our proposed SD-VAE enforces both syntactic and semantic constraints while making visually more smooth interpolation. In contrast, CVAE makes both kinds of mistakes, GVAE avoids syntactic errors but still produces semantic errors, and both methods produce subjectively less smooth interpolations.

structures.

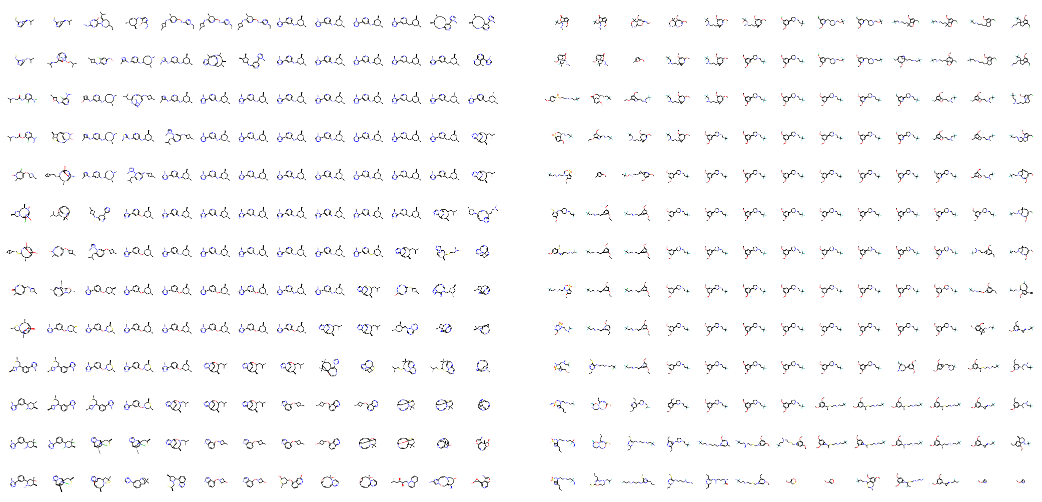


Figure 2.6: Latent Space visualization. We start from the center molecule and decode the neighborhood latent vectors (neighborhood in projected 2D space).

2.6 Conclusion

In this paper we propose a new method to tackle the challenge of addressing both syntax and semantic constraints in generative model for structured data. The newly proposed *stochastic lazy attribute* presents a the systematical conversion from offline syntax and semantic check to online guidance for stochastic generation, and empirically shows consistent and significant improvement over previous models, while requiring similar computational cost as previous model.

Our work opens the door for future works in several directions that are worthy of investigating. Firstly, we would like to explore the refinement of formalization on the more theoretical ground and investigate the application of such formalization on a more diverse set of data modality, especially following recent advances in computation linguistics provides insights of semantics modeling.

Secondly, the performance of works in our like, e.g., modeling structured data on top of human-designed formal languages with string yields representing data, depends on how good the formal language itself could be. For example, it is better for programs than molecules, because computer programs, with its formalities, are designed from scratch following programming language theories while SMILES strings for molecules are less than ideal for being designed solely for easy human entry of molecules. Along this line, following up are works concerns better semantics modeling for programs [137, 138]. On the other hand, for modeling chemical and material molecules whose formal languages are less than ideal, the community has moved forward with alternatives based on representing data as graphs [74, 168].

Chapter Appendix

2.A Grammar

2.A.1 Grammar for Program Syntax

The syntax grammar for program is a generative contest free grammar starting with ⟨*program*⟩.

⟨*program*⟩	→ ⟨*stat list*⟩
⟨*stat list*⟩	→ ⟨*stat*⟩ ‘`;`’ ⟨*stat list*⟩ | ⟨*stat*⟩
⟨*stat*⟩	→ ⟨*assign*⟩ | ⟨*return*⟩
⟨*assign*⟩	→ ⟨*lhs*⟩ ‘`=`’ ⟨*rhs*⟩
⟨*return*⟩	→ ‘`return:`’ ⟨*lhs*⟩
⟨*lhs*⟩	→ ⟨*var*⟩
⟨*var*⟩	→ ‘`v`’ ⟨*var id*⟩
⟨*digit*⟩	→ ‘`1`’ | ‘`2`’ | ‘`3`’ | ‘`4`’ | ‘`5`’ | ‘`6`’ | ‘`7`’ | ‘`8`’ | ‘`9`’
⟨*rhs*⟩	→ ⟨*expr*⟩
⟨*expr*⟩	→ ⟨*unary expr*⟩ | ⟨*binary expr*⟩
⟨*unary expr*⟩	→ ⟨*unary op*⟩ ⟨*operand*⟩ | ⟨*unary func*⟩ ‘`(`’ ⟨*operand*⟩ ‘`)`’
⟨*binary expr*⟩	→ ⟨*operand*⟩ ⟨*binary op*⟩ ⟨*operand*⟩
⟨*unary op*⟩	→ ‘`+`’ | ‘`-`’
⟨*unary func*⟩	→ ‘`sin`’ | ‘`cos`’ | ‘`exp`’
⟨*binary op*⟩	→ ‘`+`’ | ‘`-`’ | ‘`*`’ | ‘`/`’
⟨*operand*⟩	→ ⟨*var*⟩ | ⟨*immediate number*⟩
⟨*immediate number*⟩	→ ⟨*digit*⟩ ‘`.`’ ⟨*digit*⟩
⟨*digit*⟩	→ ‘`0`’ | ‘`1`’ | ‘`2`’ | ‘`3`’ | ‘`4`’ | ‘`5`’ | ‘`6`’ | ‘`7`’ | ‘`8`’ | ‘`9`’

2.A.2 Grammar for Molecule Syntax

Our syntax grammar for molecule is based on OpenSMILES standard, a context free grammar starting with ⟨*s*⟩.

⟨*s*⟩	→ ⟨*atom*⟩

⟨*smiles*⟩	→	⟨*chain*⟩
⟨*atom*⟩	→	⟨*bracket atom*⟩ | ⟨*aliphatic organic*⟩ | ⟨*aromatic organic*⟩
⟨*aliphatic organic*⟩	→	‘`B`’ | ‘`C`’ | ‘`N`’ | ‘`O`’ | ‘`S`’ | ‘`P`’ | ‘`F`’ | ‘`I`’ | ‘`Cl`’ | ‘`Br`’
⟨*aromatic organic*⟩	→	’c’ | ’n’ | ’o’ | ’s’
⟨*bracket atom*⟩	→	‘`[`’ ⟨*bracket atom (isotope)*⟩ ‘`]`’
⟨*bracket atom (isotope)*⟩	→	⟨*isotope*⟩ ⟨*symbol*⟩ ⟨*bracket atom (chiral)*⟩
	|	⟨*symbol*⟩ ⟨*bracket atom (chiral)*⟩
	|	⟨*isotope*⟩ ⟨*symbol*⟩ | ⟨*symbol*⟩
⟨*bracket atom (chiral)*⟩	→	⟨*chiral*⟩ ⟨*bracket atom (h count)*⟩
	|	⟨*bracket atom (h count)*⟩
	|	⟨*chiral*⟩
⟨*bracket atom (h count)*⟩	→	⟨*h count*⟩ ⟨*bracket atom (charge)*⟩
	|	⟨*bracket atom (charge)*⟩
	|	⟨*h count*⟩
⟨*bracket atom (charge)*⟩	→	⟨*charge*⟩
⟨*symbol*⟩	→	⟨*aliphatic organic*⟩ | ⟨*aromatic organic*⟩
⟨*isotope*⟩	→	⟨*digit*⟩ | ⟨*digit*⟩ ⟨*digit*⟩ | ⟨*digit*⟩ ⟨*digit*⟩ ⟨*digit*⟩
⟨*digit*⟩	→	’1’ | ’2’ | ’3’ | ’4’ | ’5’ | ’6’ | ’7’ | ’8’
⟨*chiral*⟩	→	‘`@`’ | ‘`@@`’
⟨*h count*⟩	→	‘`H`’ | ‘`H`’ ⟨*digit*⟩
⟨*charge*⟩	→	’-’ | ’-’ ⟨*digit*⟩ | ’+’ | ’+’ ⟨*digit*⟩
⟨*bond*⟩	→	‘−’ | ‘=’ | ‘#’ | ‘/’ | ‘\’
⟨*ringbond*⟩	→	⟨*digit*⟩
⟨*branched atom*⟩	→	⟨*atom*⟩ | ⟨*atom*⟩ ⟨*branches*⟩ | ⟨*atom*⟩ ⟨*ringbonds*⟩
	|	⟨*atom*⟩ ⟨*ringbonds*⟩ ⟨*branches*⟩
⟨*ringbonds*⟩	→	⟨*ringbonds*⟩ ⟨*ringbond*⟩ | ⟨*ringbond*⟩
⟨*branches*⟩	→	⟨*branches*⟩ ⟨*branch*⟩ | ⟨*branch*⟩
⟨*branch*⟩	→	’(’ ⟨*chain*⟩ ’)’ | ’(’ ⟨*bond*⟩ ⟨*chain*⟩ ’)’
⟨*chain*⟩	→	⟨*branched atom*⟩ | ⟨*chain*⟩ ⟨*branched atom*⟩
	|	⟨*chain*⟩ ⟨*bond*⟩ ⟨*branched atom*⟩

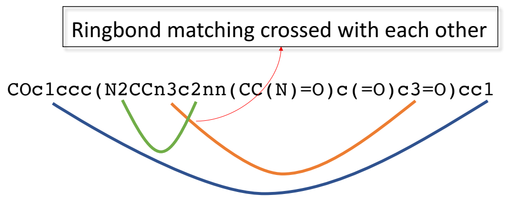

Ringbond matching crossed with each other
COc1ccc(N2CCn3c2nn(CC(N)=O)c(=O)c3=O)cc1

Figure 2.7: Example of cross-serial dependencies (CSD) that exhibits in SMILES language.

2.A.3 Examples of SMILES semantics

Here we provide more explanations of the semantics constraints that contained in SMILES language for molecules.

Specifically, the semantics we addressed here are:

1. **Ringbond matching:** The ringbonds should come in pairs. Each pair of ringbonds has an index and a bond-type associated. What the SMILES semantics requires is exactly the same as the well-known cross-serial dependencies (CSD) in formal language. CSD also appears in some natural languages, such as Dutch and Swiss-German. Another example of CSD is a sequence of multiple different types of parentheses where each separately balanced disregarding the others. See Figure 2.7 for an illustration.

2. **Explicit valence control:** Intuitively, the semantics requires that each atom cannot have too many bonds associated with it. For example, a normal carbon atom has maximum valence of 4, which means associating a Carbon atom with two triple-bonds will violate the semantics.

2.A.4 Dependency graph introduced by Attribute Grammar

Suppose there is a production $r = u_0 \to u_1 u_2 \dots u_{|\beta|} \in \mathcal{R}$ and an attribute $u_i.a$ we denote the dependency set

$$D^r(u_i.a) = \{u_j.b | u_j.b \text{ is required for calculating } u_i.a\}.$$

The union of all dependency sets $\mathcal{D}_{\mathcal{T}}^{(att)} = \bigcup_{r \in \mathcal{T}, u_i \in r} D^r(u_i.a)$ induces a dependency graph, where nodes are the attributes and directed edges represents the dependency relationships between those attributes computation. Here $\mathcal{T}$

is an (partial or full) instantiation of the generated syntax tree of grammar G. Let $D^r(u_i) = \{u_j | \exists a, b : u_j.b \in D^r(u_i.a)\}$ and $\mathcal{D}_\mathcal{T} = \bigcup_{r \in \mathcal{T}, u_i \in r} D^r(u_i)$, that is, $\mathcal{D}_\mathcal{T}$ is constructed from $\mathcal{D}_\mathcal{T}^{(att)}$ by merging nodes with the same symbol but different attributes, we call $\mathcal{D}_\mathcal{T}^{(att)}$ is noncircular if the corresponding $\mathcal{D}_\mathcal{T}$ is noncircular.

In our paper, we assume the noncircular property of the dependency graph. Such property will be exploited for top-down generation in our decoder.

2.B Training Details

Since our proposed SD-VAE differentiate itself from previous works (CVAE, GVAE) on the formalization of syntax and semantics, we therefore use the same deep neural network model architecture for a fair comparison. In encoder, we use 3-layer one-dimension convolution neural networks (CNNs) followed by a full connected layer, whose output would be fed into two separate affine layers for producing μ and σ respectively as in reparameterization trick; and in decoder we use 3-layer RNNs followed by a affine layer activated by softmax that gives probability for each production rule. In detail, we use 56 dimensions the latent space and the dimension of layers as the same number as in [85]. As for implementation, we use [85]'s open sourced code for baselines, and implement our model in PyTorch framework [5].

In a 10% validation set we tune the following hyper parameters and report the test result from setting with best valid loss. For a fair comparison, all tunings are also conducted in the baselines.

We use $\texttt{ReconstructLoss} + \alpha\texttt{KLDivergence}$ as the loss function for training. A natural setting is $\alpha = 1$, but [85] suggested in their open-sourced implementation[6] that using $\alpha = 1/\texttt{LatentDimension}$ would leads to better results. We explore both settings.

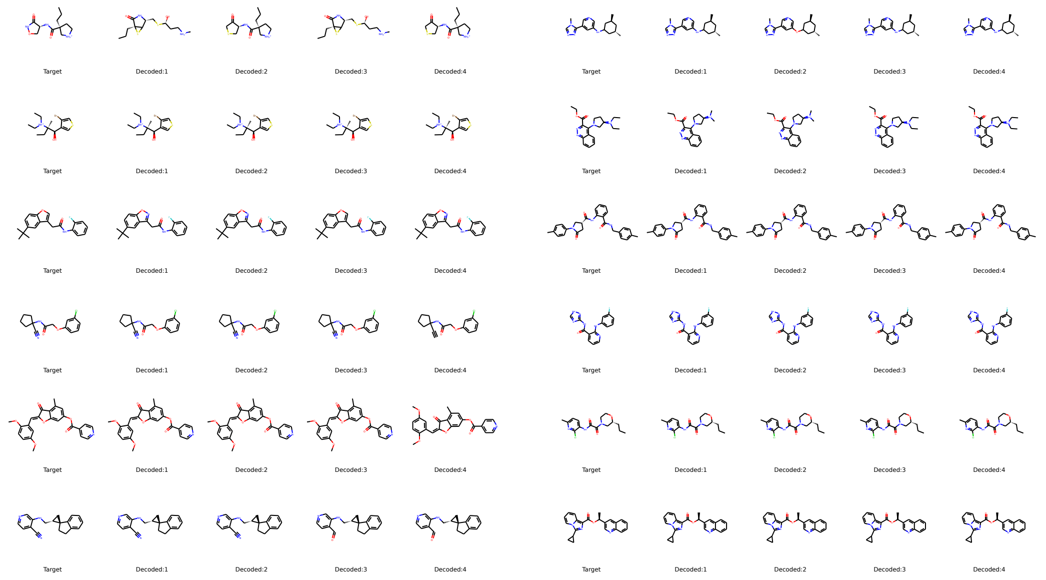


Figure 2.8: Visualization of reconstruction. The first column in each figure presents the target molecules. We first encode the target molecules, then sample the reconstructed molecules from their encoded posterior.

2.C More experiment details

2.C.1 Bayesian Optimization

The Bayesian optimization is used for searching latent vectors with desired target property. For example, in symbolic program regression, we are interested in finding programs that can fit the given input-output pairs; in drug discovery, we are aiming at finding molecules with maximum drug likeness. To get a fair comparison with baseline algorithms, we follow the settings used in [85].

Specifically, we first train the variational autoencoder in an unsupervised way. After obtaining the generative model, we encode all the structures into latent space. Then these vectors and corresponding property values (*i.e.*, estimated errors for program, or drug likeness for molecule) are used to train a sparse Gaussian process with 500 inducing points. This is used

[5]`http://pytorch.org/`

[6]`https://github.com/mkusner/grammarVAE/issues/2`

later for predicting properties in latent space. Next, 5 iterations of batch Bayesian optimization with the expected improvement (EI) heuristic is used for proposing new latent vectors. In each iteration, 50 latent vectors are proposed. After the proposal, the newly found programs/molecules are then added to the batch for next round of iteration.

During the proposal of latent vectors in each iteration, we perform 100 rounds of decoding and pick the most frequent decoded structures. This helps regulates the decoding due to randomness, as well as increasing the chance for baselines algorithms to propose valid ones.

2.C.2 Recontruction

We visualize some reconstruction results of SMILES in Figure 2.8. It can be observed that, in most cases the decoder successfully recover the exact origin input. Due to the stochasticity of decoder, it may have some small variations.

3 Social Relation Inference via Label Propagation [7]

3.1 Introduction

In collaboration networks, edges, or social relations [149], are formed between people with shared interests. Social relations in networks are complex and nuanced, which often cannot be characterized by a single label. Consider a co-author network between researchers where the social relations between two researchers are the research areas they collaborate in. Since collaborations can occur in different research areas, the social relation between researchers is inherently multifaceted. Many applications on collaboration networks can benefit from an awareness of social relations, such as node classification [165], recommendation [144] and anomaly detection [162]. However, in many networks, such label information (social relations) is far from complete. It is thus desirable to learn to infer social relations associated with the unlabeled edges.

We formalize the task of social relation inference as a semi-supervised multi-label edge classification problem on networks. Given the network structure and a limited amount of labeled edges, our goal is to infer the labels of the rest of the edges. There are several previous studies on inferring social ties from social networks, which is similar to our definition of social relations [144, 142]. However, these works assume that each edge corresponds to a single relation type, which may not be the case in collaboration networks. Moreover, they only consider first-order or second-order relationships between nodes, but fails to model higher-order relationships that play an important role in network inference tasks [24].

Another relevant area is network embeddings [120, 141, 57], which aim at learning low-dimensional latent representations of nodes in a network. Also, representations of larger-scale components of networks (such as edges and subgraphs) can be composed from these node representations. These representations can then be used as features for a wide range of downstream

[7]The content of this section is taken from:
[146] Yingtao Tian, Haochen Chen, Bryan Perozzi, Muhao Chen, Xiaofei Sun, and Steven Skiena. Social relation inference via label propagation. In *Proceedings of the 41st European Conference on Information Retrieval, ECIR 2019, Cologe, Germany, April 14-18, 2018*, 2019

tasks on networks, including social relation inference. As a pioneering work, DeepWalk [120] generates fixed-length random walk sequences in networks and trains a skip-gram model [109] on these sequences to obtain node embeddings. While achieving state-of-the-art results on a handful of network inference tasks such as node classification and link prediction [120, 57], the semantics of edges in networks are seldom exploited by network embedding models. Moreover, we find that they usually ignore the unique properties possessed by different types of networks and by different downstream tasks. Also, many of them are computationally expensive: learning network embeddings of a one-million node network can take several days on a single CPU.

In this paper, we propose a simple but effective method for social relation inference on collaboration networks. Our method is based on the observation that social relations between people in collaboration networks are determined by their shared interests. As such, the networks are highly homophilous and there is a natural connection between the (hidden) labels of the nodes, and the provided edge labels. Using this relationship, we first transform the edge labels into a node labeling. Next, to alleviate any data sparsity problem, we perform label propagation on the input network to obtain label distribution for all nodes. Label propagation [174, 154] represents a class of semi-supervised learning methods which find numerous applications in graph mining. For social relation inference, we find that label propagation has several desirable properties compared to the neural methods mentioned before: it is extremely efficient and it makes good use of the high level of homophily exhibited in collaboration networks [121]. Finally, once node labels have been obtained, the label distribution of edges can be easily inferred from the label distribution of their endpoints. Experimental results on real-world networks show that our method outperforms state-of-the-art methods by a large margin.

3.2 Problem Definition and Notation

We hereby formalize the problem of social relation inference in collaboration networks. Let $G = (V, E)$ be an undirected graph, where V are the nodes in the graph and E represent its edges. Let A be the adjacency matrix of G. Let $L = (l_1, l_2, \cdots, l_k)$ be the set of relation types (labels). A partially labeled network is then defined as $G = (V, E_L, E_U, Y_L)$, where E_L is the set of labeled edges, E_U is the set of unlabeled edges with $E_L \cup E_U = E$. Y_L

Algorithm 2 LabelProp(G, P)

```
Input: graph $G$, initial node label distribution $P$, rounds of iteration $k$
Output: node label distribution after propagation $\hat{Y_V} \in \mathbb{R}^{|V|\times|L|}$
1: Compute the degree matrix $D$: $D_{ii} \leftarrow \sum_j A_{ij}$
2: Compute the transition matrix: $Q \leftarrow D^{-1}A$
3: $Y^{(0)} \leftarrow P$
4: for $i = 0$ to $k-1$ do
5:     $Y^{(i+1)} \leftarrow QY^{(i)}$
6: end for
7: $\hat{Y_V} = Y^{(k)}$
8: return $\hat{Y_V}$
```

represents the relation types associated with the labeled edges in E_L, with $\forall Y_L(i) \in Y_L : Y_L(i) \subseteq L$. The objective of social relation inference is to predict the relation types Y_U of the unlabeled edges E_U:

$$f : G = (V, E_L, E_U, Y_L) \rightarrow Y_U \tag{3}$$

We denote the i-th row and ij-th element of a matrix M as M_i and M_{ij}.

3.3 Method

3.3.1 Step 1: From Edge Labels to Node Labels

One challenge with social relation inference is that the labels we seek to predict are associated with edges, instead of nodes. However, most machine learning algorithms on graphs only operate on nodes. To bridge this gap, we note that collaboration networks possess a unique property: edges are typically formed between two people which have *shared interests*. Such shared interests can very well be characterized by the labels of edges. This means that we should be able to infer the latent interests of nodes based on their corresponding edge labels.

Formally, we seek to estimate the probability distribution matrix $P \in \mathbb{R}^{|V|\times|L|}$ for all nodes over the label space L. For ease of presentation, we assume that the training data is given in the form of triplets $t = (u, v, l)$, where $u, v \in V, l \in L$. In other words, if an edge has several labels, then we construct one triplet for each label. We define the set of all training triplets as T. Assume the label distribution of u and v are independent, the strength of relation l between u and v can be estimated as:

$$Pr(l|u,v) = P_{ul} \cdot P_{vl} \tag{4}$$

Our objective is to maximize the probability of observing the relations in T as given by:

$$\ell = \prod_{u \in V} \prod_{\substack{(v,l) \\ (u,v,l) \in T}} Pr(l|u,v) \tag{5}$$

Then, for a certain $u \in V$, our goal is to minimize the following objective:

$$-\log \ell_u = -\sum_{\substack{(v,l) \\ (u,v,l) \in T}} (\log P_{ul} + \log P_{vl}) \tag{6}$$

Since P is the probability distribution of labels, we have the constraint $\sum_{l \in L} P_{ul} = 1$. The Lagrangian function of Eq. (6) is:

$$\mathcal{L}(P_u, \lambda) = -\sum_{\substack{(v,l) \\ (u,v,l) \in T}} (\log P_{ul} + \log P_{vl}) + \lambda(\sum_{l \in L} P_{ul} - 1) \tag{7}$$

For all $l \in L$, we take the derivative of Eq. 7 w.r.t. P_{ul} and set it to zero:

$$-\frac{\#(u,l)}{P_{ul}} + \lambda = 0 \tag{8}$$

where $\#(u,l)$ is the number of co-occurrences of u and l in T, with v being marginalized out. It is now clear that $P_{ul} = \frac{\#(u,l)}{\lambda}$. Combined with the constraint $\sum_{l \in L} P_{ul} = 1$, we have $\lambda = \sum_{l \in L} \#(u,l)$. Finally, the closed-form estimation of P_{ul} is calculated as: $P_{ul} = \#(u,l) / \sum_{l \in L} \#(u,l)$.

Concretely, we can simply compute the relative frequency that each node co-occur with each label, which gives us the initial label distribution P of all nodes.

3.3.2 Step 2: Label Propagation

Labeled edges are often scarce in real-world collaboration networks. As a result, using the procedure outlined above, we may get an empty label distribution for most of the nodes (as they have no edges). To alleviate this

Table 3.1: Statistics of the networks used in our experiments.

Dataset	# Vertices	# Edges	# Train	# Test	# Valid	# Classes
Arnet-Small	187,939	1,619,278	1,579,278	20,000	20,000	100
Arnet-Medium	268,037	2,747,386	2,147386	300,000	300,000	500
Arnet-Large	945,589	5,056,050	3,856,050	600,000	600,000	500

problem, we propose using label propagation [174] on G to spread the information from labeled edges around the graph. Algorithm 2 details the process. We start from the initial label distribution obtained in Step 1 and repeatedly distribute node labels to the neighboring nodes.

3.3.3 Step 3: From Node Labels to Edge Labels

Once we have obtained the label distribution for all nodes, we can easily compute the label distribution for edges by reusing Eq. 4. For each edge $e = (u, v)$, the strength of relation l is $P_{ul} \cdot P_{vl}$. The ranking of relation strengths serves as our prediction of social relations.

3.3.4 Time Complexity Analysis

The majority of time complexity is contributed by Algorithm 2, which takes $O(k \cdot (|E| + |V| \cdot |L|))$. In our experiments, it is further shown that a small value of k is sufficient for our model to converge: empirically, we take $k = 5$ based on the performance on the validation set. We provide detailed running time comparison against baseline methods in Section 3.4.

3.4 Experiment

In this section, we describe the datasets for social relation inference and compare our method against a number of baselines.

3.4.1 Dataset

We use the processed ArnetMiner [143] datasets provided by TransNet [149]. ArnetMiner is a large-scale co-author network with over a million authors and four million collaboration relations. The social relations between researchers can be reflected by the research areas or topics they collaborate in.

Concretely, for each co-author relationship, the authors of TransNet extract representative research interest phrases from the abstracts of co-authored papers as edge labels. Two collaboration networks of different scales and different amount of labels are provided in this dataset to better investigate the characteristics of different models. We use the same data split as in TransNet [149]. The statistics of the datasets are presented in Table 3.1.

3.4.2 Baseline Methods

The baseline methods we use are as follows: **(1) DeepWalk** [120]: This is a network embedding method that learns latent representations of nodes in a graph. **(2) LINE** [141]: This is a network embedding method that preserves both first-order and second-order proximities in networks. **(3) node2vec** [57]: This is a network embedding method that improves DeepWalk with a biased random walk phase. **(4) TransE** [20]: This is a knowledge base embedding method which simultaneously learns latent representations of nodes and relations. Since TransE models each relation separately, we split each edge with k labels into k training instances, one for each label. **(5) TransNet** [149]: This method is an extension to TransE which explicitly models edges with multiple labels. It is also the state-of-the-art method for social relation inference.

We follow the experimental setup as in TransNet [149]. For all baseline methods, we use the hyperparameter settings as described in their papers. For TransE, we use the similarity-based method to predict social relations as described in [20]. For TransNet, we follow the inference algorithm in their paper. For the three network embedding methods, we concatenate node representations as the feature vector for edges. For social relation inference, we train a one-vs-rest logistic regression model with L2 regularization implemented in LibLinear [51].

3.4.3 Results and Analysis

In Tables 3.2 and 3.3, we summarize the experimental results using the same data split as TransNet. Results for all baseline methods (including TransNet) are taken from the TransNet paper. We can clearly see that our simple method outperforms all baseline methods by a large margin. The performance gain over the best baseline method, TransNet, is at least 3.5% and up to 8.4% in terms of hits@5. We note that the TransNet data split uses

Table 3.2: Relation inference results on **Arnet-Small**.

Algorithm	**Metrics(%)**		
	hits@1	*hits*@5	*hits*@10
DeepWalk	13.88	36.80	50.57
LINE	11.30	31.70	44.51
node2vec	13.63	36.60	50.27
TransE	39.16	78.48	88.54
TransNet	47.67	86.54	92.27
Proposed	**48.89**	**90.13**	**93.90**

Table 3.3: Relation inference results on **Arnet-Large**.

Algorithm	**Metrics(%)**		
	hits@1	*hits*@5	*hits*@10
DeepWalk	5.41	16.17	23.33
LINE	4.28	13.44	19.85
node2vec	5.39	16.23	23.47
TransE	15.38	41.87	55.54
TransNet	28.85	66.15	75.55
Proposed	**29.91**	**72.32**	**80.86**

98%, 76% and 78% edges as training data for Arnet-Small, Arnet-Medium and Arnet-Large respectively. With such a large amount of training data, our algorithm achieves the reported performance even without performing label propagation, which proves the effectiveness of the node label inference algorithm. Moreover, our algorithm is orders of magnitude faster than all baseline methods. Using a single CPU core at 2.0GHz, our method finishes in 5 minutes on Arnet-Small while all baseline methods take more than 24 hours.

The only hyperparameter in our algorithm is the number of rounds of iterations k for label propagation, which is tuned on the validation set. We observe that even with only 1% of labeled edges, our label propagation algorithm converges within 5 iterations.

3.5 Conclusion

We study the problem of inferring social relations in collaboration networks, formulated as a semi-supervised learning problem on graphs where edges have multiple labels. Observing that edges in collaboration networks represent the shared interests of two people, we transform edge labels to node labels and perform label propagation to deal with the label sparsity problem. Experimental results on real-world collaboration networks show the superiority of our method in terms of both accuracy and efficiency.

It is interesting to see a straightforward approach like our work archive state-of-the-art performance. We hypothesize that this is due to the type of data represented in the social network where attributes are implicitly associate also with nodes besides edges, which benefits our propagation-based approach centered at aggregated attributes on nodes. The future work calls for exploring principled indicator that characterizes a class of datasets that can benefit from this simple approach, as well as theoretical analysis of the advantages brought by our approaches. Furthermore, we would also like to investigate proper approaches to handle datasets that this work may not be ideal for, such as ones where relations are not strongly associate with nodes.

4 Towards the High-quality Anime Characters Generation with Generative Adversarial Networks [8]

4.1 Introduction

The automatic generation of anime characters offers an opportunity to bring a custom character into existence without professional skill. Besides, professionals may also take advantages of the automatic generation for inspiration on animation and game character design. However results from existing models [106, 127, 145, 92] on anime image generation are blurred and distorted on a non-trivial frequency, thus generating industry-standard facial images for anime characters remains a challenge. In this paper, we propose a model that produces anime faces at high quality with a promising rate of success with three-fold contributions: A clean dataset from Getchu, a suitable DRAGAN [82]-based SRResNet [91]-like GAN model and our general approach to training a conditional model from images with estimated tags as conditions. We also make available a publicly accessible web interface.

Our contribution is best highlighted using high-quality high-resolution (256 by 256) images sampled generated from our model:

Randomly Generated Examples: Figure 4.1 shows images generated from our model where both noise and attributes are randomly sampled.

Generated Examples with Fixed Noise: In Figure 4.2 we show that by fixing the random noise part and sampling random attribution, the model generates images that have similar major visual features like face shapes and directions, evidence of the generalization ability.

Generated Examples with Fixed attribution: In Figure 4.3 we show generated images from fixed attribution and randomly noise. The model here generates images with desired attributions but with different, variant visual features.

Interpolation between images: In Figure 4.4 we show the interpolation between two sets of randomly selected features. It shows that attributes, like noise, are meaningful under the continuous setting.

[8]The content of this section is taken from:
[75] Yanghua Jin, Jiakai Zhang, Minjun Li, Yingtao Tian, Huachun Zhu, and Zhihao Fang. Towards the automatic anime characters creation with generative adversarial networks. *CoRR*, abs/1708.05509, 2017

Details of protocol and more examples can be found in Section 4.4.

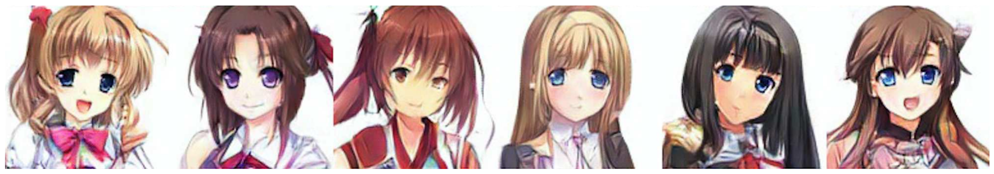


Figure 4.1: Generated samples with random prior.

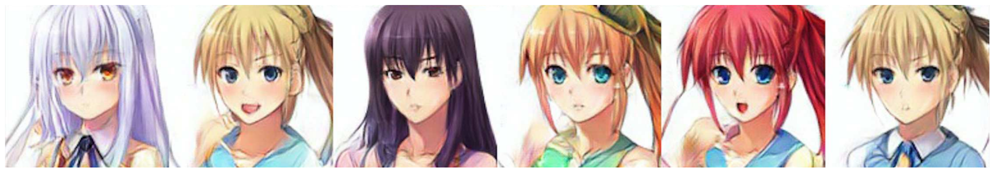


Figure 4.2: Generated images with fixed noise part and random attributes.

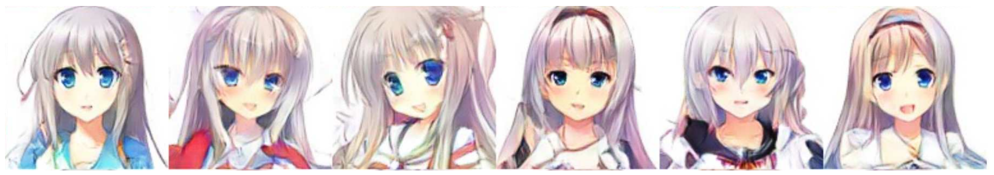


Figure 4.3: Generated images with fixed conditions (silver hair, long hair, blush, smile, open mouth, blue eyes) and random noise part.

4.2 Dataset Construction

We propose to use a consistent, clean, high-quality anime dataset collected from Getchu, a website for Japanese games. We describe the process in detail in Section 4.2.1). The generation of images with customization requires categorical metadata of images in priors as attributes along with noise. Since Getchu does not provide such metadata, we use Illustration2Vec [132], a CNN-based tool for anime illustrations that serves as attribute estimation. We show the details of attribute estimation with its statistics and visualization in Section 4.2.2.

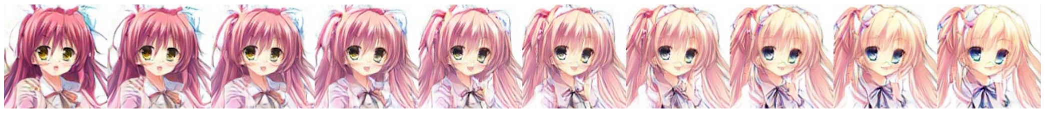


Figure 4.4: Interpolation between images.

4.2.1 Image Collection

Getchu[9] is a website providing information and selling of Japanese games, for which there are character introduction sections with standing pictures. Figure 4.5 shows one sample character introduction from the site. These images are diverse enough since illustrators create them with different styles for games in a diverse set of themes, yet consisting since they are all belonging to the domain of character images, are in decent quality, and are appropriately clipped/aligned due to the nature of illustration purpose. Because of these properties, they are suitable for our task.

Then we download images and apply animeface [10], an anime character face detector, to each image and get the bounding box for faces and landmarks for facial features such as mouth and eyes.

We observe that using estimated bounding box suffers from the fact that many faces are too close to capture complete character attributes, and are often rotated such that they are not well-aligned vertically or horizontally. To tackle this issue, inspired by preprocessing in [77] we estimate a more reliable, possibly rotated, face frame from the facial landmarks. In detail, our inference of face frame from facial landmarks is the following:

[9] www.getchu.com

[10] https://github.com/nagadomi/animeface-2009

[11] Getchu page with standing picture (http://www.getchu.com/soft.phtml?id=933144). Copyright: Frontwing, 2017.

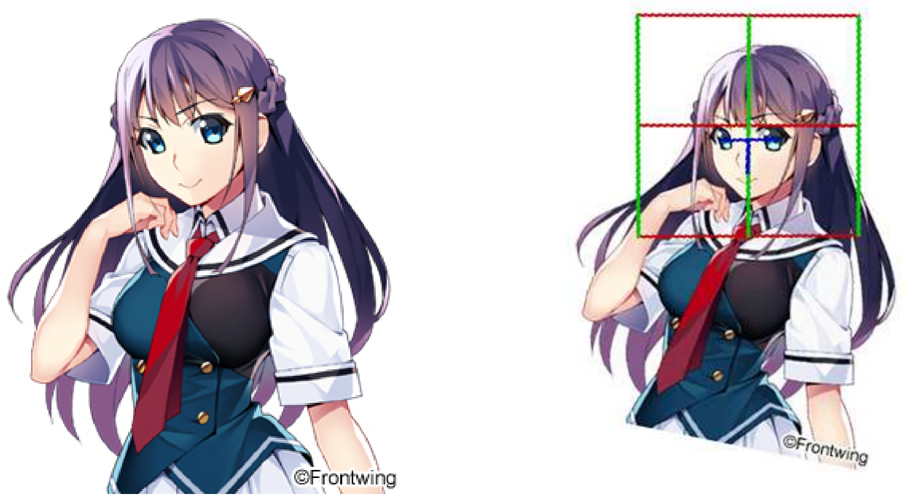


Figure 4.5: Example of Image processing using face frame detection and rotation. The original image of standing picture on the left[11], and detected face frame on the right shown with rotation applied. The green and red lines indicate the face frame and based on the landmarks of two eyes and mouth indicated by the blue “T”-shape.

$$\begin{aligned} x' &= e_1 - e_0 \\ e_c &= \frac{1}{2}\left(e_0 + e_1\right) \\ y' &= e_c - m \\ c &= e_c + 0.4y' \\ s &= \max\left(4.0 \cdot |x'|, 5.2 \cdot |y'|\right) \\ x &= \mathrm{Normalize}\left(x' - \mathrm{Rotate90}\left(y'\right)\right) \\ y &= \mathrm{Rotate90}\left(x\right) \end{aligned}$$

where e_0, e_1 and m represent the 2-d pixel location of two eyes and mouth, respectively. c and s are for center and size of the face frame, and x, y indicate the orientation of the frame. We pad white pixels to extend the size of images when the frame may include regions outside the original image. this pipeline is illustrated in Figure 4.5, and use faces defined by these frames as dataset.

Finally, from 42000 face images in total inferred from face frames, we manually check all anime face images and remove about 4% false positive and undesired images.

4.2.2 Tag Estimation

blonde hair 4991	brown hair 6659	black hair 4842	blue hair 3289	pink hair 2486	purple hair 2972	green hair 1115
red hair 2417	silver hair 987	white hair 573	orange hair 699	aqua hair 168	gray hair 57	long hair 16562
short hair 1403	twintails 5360	drill hair 1683	ponytail 8861	blush 4926	smile 5583	open mouth 4192
hat 1403	ribbon 5360	glasses 1683	blue eyes 8861	red eyes 4926	brown eyes 5583	green eyes 4192
purple eyes 4442	yellow eyes 1700	pink eyes 319	aqua eyes 193	black eyes 990	orange eyes 49	

Table 4.1: Number of dataset images for each tag

To overcome the limitation that images collected from Getchu are without any tag, we use Illustration2Vec [132], a pre-trained[12] CNN-based tool for estimating tags of anime illustrations. Given an anime image, this network can predict probabilities of belonging to 512 kinds of general attributes (tags) such as “smile” and “weapon”, among which we select 34 related tags suitable for our task. We show the selected tags and the number of dataset images corresponded to each estimated tag in Table 4.1. For the set of tags with mutual exclusivity (e.g., hair color, eye color), we choose the one with maximum probability from the network as the estimated tag. For orthogonal tags (e.g. “smile”, “open mouth”, “blush”), we use 0.25 as the threshold and estimate each attribute’s presence independently.

We would like to show the image preparation and the performance of tag estimation through visualization. As an approximation, we apply the Illustration2Vec feature extractor, which largely shares architecture and weights with Illustration2Vec tag estimator, on each image for a 4096-dimension feature vector, and project feature vectors onto a 2D space using t-SNE[105].

[12]Pre-trained model available on http://illustration2vec.net/

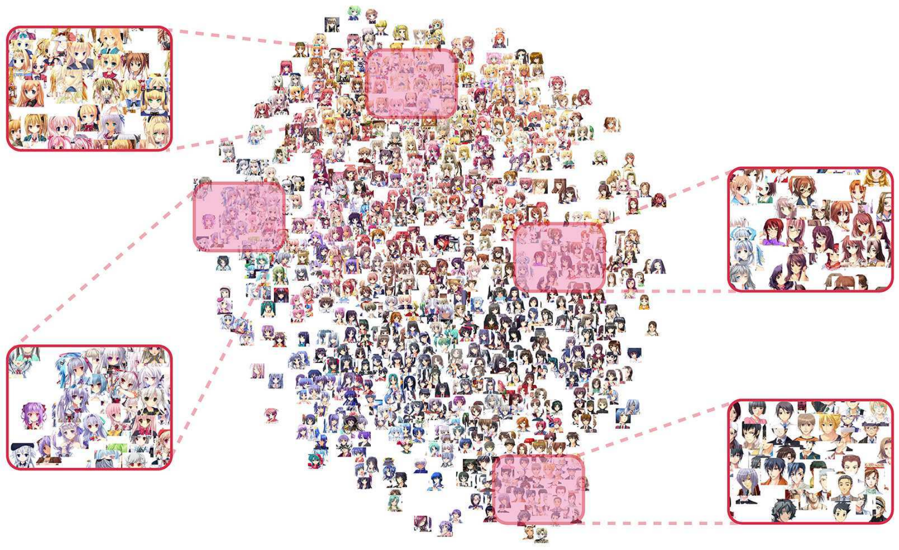


Figure 4.6: t-SNE visualization of 1500 dataset images. A clustering in terms of similar attributes can be observed in close-up views.

Figure 4.6 shows the t-SNE result of 1500 images sampled from the dataset. We observe that character images with similar visual attributes are placed closely. Due to the shared weights, we believe this also indicates the good performance in tag estimator.

4.3 Generative Model

Generative Adversarial Networks proposed by Goodfellow et at.[55] are implicit generative models. It proves to be an effective and efficient way to generate highly photo-realistic images in an unsupervised and likelihood-free manner[122]. GAN uses a generator network G to generate samples from P_G. This is done by transforming a latent noise variable $z \sim P_{noise}$ into a sample $G(z)$. The original GAN uses a min-max game strategy to train the generator G, imposing another network D to distinguish samples from G and real samples. Formally, the objective of GAN can be expressed as

$$\min_G \max_D \mathcal{L}(D, G) = \mathbb{E}_{x \sim P_{data}}[\log D(x)] + \mathbb{E}_{x \sim P_{noise}}[\log(1 - D(G(z)))] \,.$$

In this formula, the discriminator D try to maximize the output confidence score from real samples. Meanwhile, it also minimizes the output confidence score from fake samples generated by G. In contrast, G aims to maximize the D evaluated score for its outputs, which can be viewed as deceiving D.

Despite the impressive results of GAN, it is notoriously hard to train GAN properly. [6] showed the P_G and P_{data} might have non-overlap supports, so the Jensen-Shannon Divergence in the original GAN objective is constantly 0, which leads to instability. [8] argued that there might exist no equilibrium in the game between generator and discriminator. One possible remedy is to use integral probability metric(IPM) based methods instead, e.g. Wasserstein distance[7], Kernel MMD[94], Cramer distance[13]. Some recent GAN variants suggest using gradient penalty to stabilize GAN training[59, 13, 130, 82]. Mattya[107] compared several recent GAN variants under the same network architecture and measures their performance under the same metric.

Here, we use DRAGAN proposed by [82] as the basic GAN model. As [107] shows, DRAGAN can give presumable results compare to other GANs, and it has the least computation cost among those GAN variants. Compare with Wasserstein GAN and its variants, DRAGAN can be trained under the simultaneous gradient descent setting, which makes the training much faster. In our experiments, we also find it is very stable under several network architectures, and we successfully train the DRAGAN with an SRResNet [91]-like generator, Incorporating label information is important in our task to provide the user a way to control the generator. Inspired by ACGAN [118], we utilize the attributions by feeding them along with noise vector and add a multi-label classifier on the top of the discriminator for reconstructing the attributions.

In the rest of this section, we describe the network architecture, loss function, and training details.

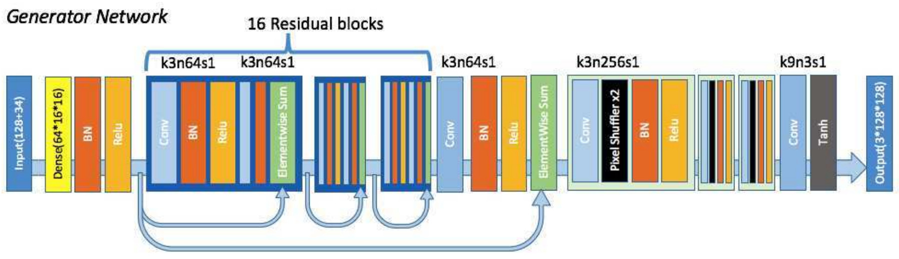


Figure 4.7: Generator Architecture

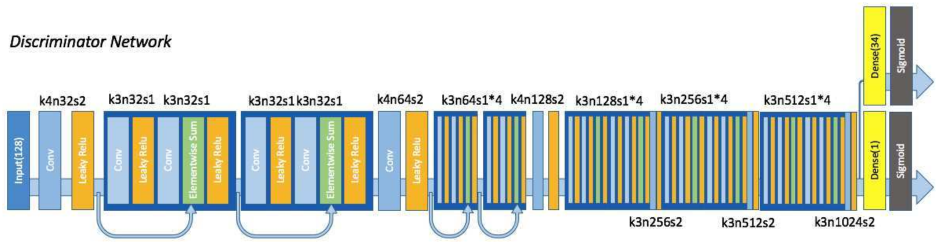


Figure 4.8: Discriminator Architecture

Network Architecture The generator's architecture is shown in Figure 4.7, which is a modification from SRResNet [91]. The model contains 16 ResBlocks and uses 3 sub-pixel CNN [135] for feature map upscaling. Figure 4.8 shows the discriminator architecture, which contains 10 ResBlocks in total. All batch normalization layers are removed in the discriminator since it would bring correlations within the mini-batch, which is undesired for the computation of the gradient norm. We add an extra fully-connected layer to the last convolution layer as the attribute classifier. All weights are initialized from a Gaussian distribution with mean 0 and standard deviation 0.02.

Loss Function The loss function is described as following:

$$\begin{aligned}
\mathcal{L}_{adv}(D) &= -\mathbb{E}_{x\sim P_{data}}[\log D(x)] - \mathbb{E}_{x\sim P_{noise}, c\sim P_{cond}}[\log(1 - D(G(z,c)))] \\
\mathcal{L}_{cls}(D) &= \mathbb{E}_{x\sim P_{data}}[\log P_D[label_x|x]] + \mathbb{E}_{x\sim P_{noise}, c\sim P_{cond}}[\log(P_D[c|G(z,c)])] \\
\mathcal{L}_{gp}(D) &= \mathbb{E}_{\hat{x}\sim P_{perturbed_data}}[(||\nabla_{\hat{x}} D(\hat{x})||_2 - 1)^2] \\
\mathcal{L}_{adv}(G) &= \mathbb{E}_{x\sim P_{noise}, c\sim P_{cond}}[\log(D(G(z,c)))] \\
\mathcal{L}_{cls}(G) &= \mathbb{E}_{x\sim P_{noise}, c\sim P_{cond}}[\log(P_D[c|G(z,c)])] \\
\mathcal{L}(D) &= \mathcal{L}_{cls}(D) + \lambda_{adv}\mathcal{L}_{adv}(D) + \lambda_{gp}\mathcal{L}_{gp}(D) \\
\mathcal{L}(G) &= \lambda_{adv}\mathcal{L}_{adv}(G) + \mathcal{L}_{cls}(G)
\end{aligned}$$

where P_{cond} indicates the prior distribution of assigned tags. $\lambda_{adv}, \lambda_{gp}$ are balance factors for the adversarial loss and gradient penalty respectively.

Training details We find that the model achieves the best performance with λ_{adv} equaling to the number of attributes, a detailed analysis [172] of the gradient in the condition of ACGAN shows. Here, we set λ_{adv} to 34 and λ_{gp}

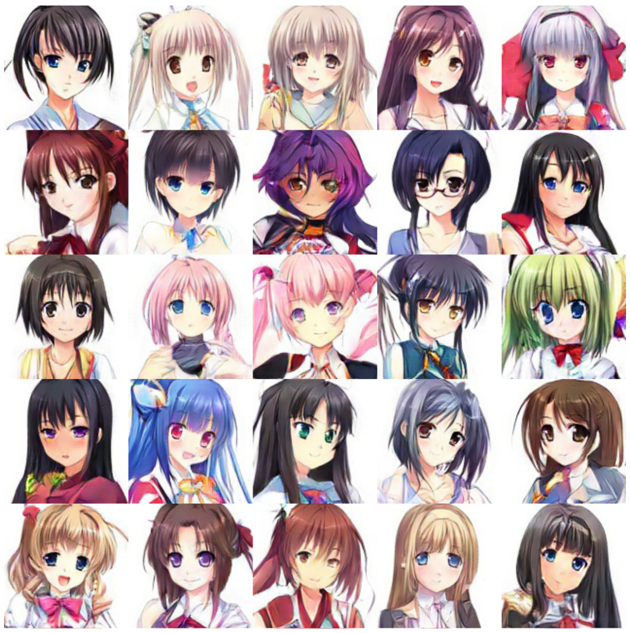


Figure 4.9: Generated samples with random noise and attributes.

to 0.5 in all experiments. All models are optimized using Adam optimizer[78] with β_1 equaling 0.5. We use a batch size of 64 in the training procedure. The learning rate is initialized to 0.0002 and exponentially decreases after 50000 iterations of training. We train our GAN model using only images from games released after 2005 and with scaling all training images to a resolution of 128 by 128 pixels. This gives 31255 training images in total. We use the following simple strategy to sample related attributes for the noise. For the categorical attributes (e.g., hair and the eye color), we randomly select one possible color with uniform distribution. For other attributes, we set each label independently with a probability of 0.25.

4.4 Qualitative Evaluation

Figure 4.9 shows more images generated from our model. Figure 4.10 is an example of fixing the random noise part and sampling random attributes, where the model can generate images that have similar major visual features. In sampling random attributes, we use the following simple strategy: For the categorical attributes (hair and the eye color), we randomly select one possi-

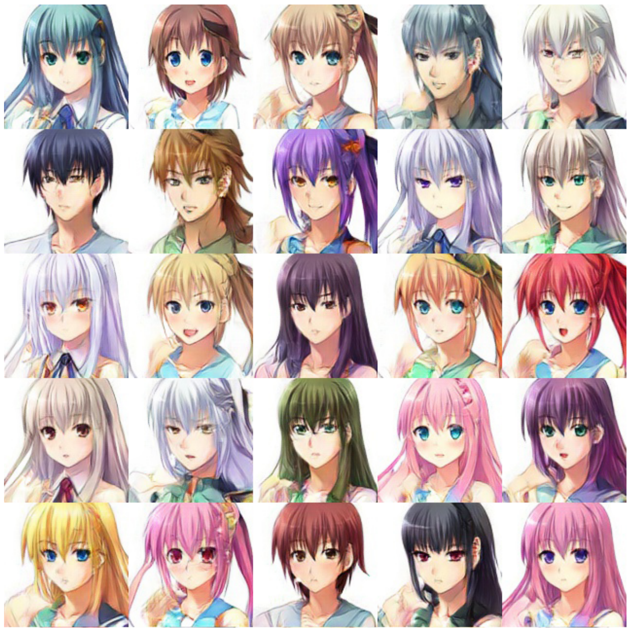


Figure 4.10: Generated images with fixed noise and random attributes.

ble attribute uniformly. For other attributes, we set each label independently with a probability of 0.25.

We empirically observe that the random noise part heavily inference the quality of the final result. Some noise vector can give good samples no matter what conditioned on, while some other noise vectors are easier to produce distorted images. As Table 4.1 states, labels are not evenly distributed in our training dataset, which results that some combinations of attributes cannot give good images. In Figure 4.11, (a)(b) are generated with well-learned attributes like "blonde hair", "blue eyes". In contrast, (c) and (d) are associated with "glasses", "drill hair", which is not well learned because of the insufficiency of corresponding training images. All characters in (a)(b) appear to be attractive, but most characters in (c)(d) are distorted.

We also show more interpolations in Figure 4.12. Although label controlling variables are assigned with discrete values in the training stage, the result shows that those discrete attributes are still meaningful under the continuous setting.

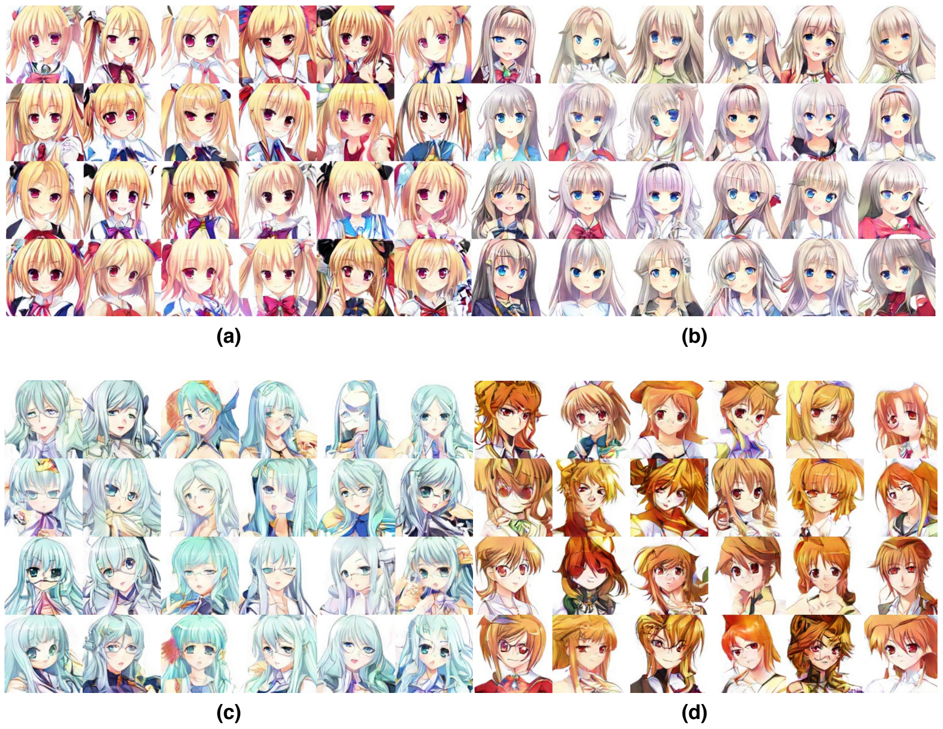


Figure 4.11: Generated images with random noise and following fixed attributes: (a) blonde hair, twintails, blush, smile, ribbon, red eyes (b) silver hair, long hair, blush, smile, open mouth, blue eyes (c) aqua hair, long hair, drill hair, open mouth, glasses, aqua eyes (d) orange hair, ponytail, hat, glasses, red eyes, orange eyes

4.5 Quantitative Evaluation

For quantitively evaluating our generate examples, we conduct *attribute precision* (Section 4.5.1), *FID evaluation* (Section 4.5.2) and *nearest training examples* (Section 4.5.3) that shows the superiority of our model over DCGAN [122] baseline.

4.5.1 Attribute Precision

To evaluate how each tag affects the output result, we measure the precision of the output result when the certain label is assigned. With each target, we fix the target label to true, and sample other labels in random. For each

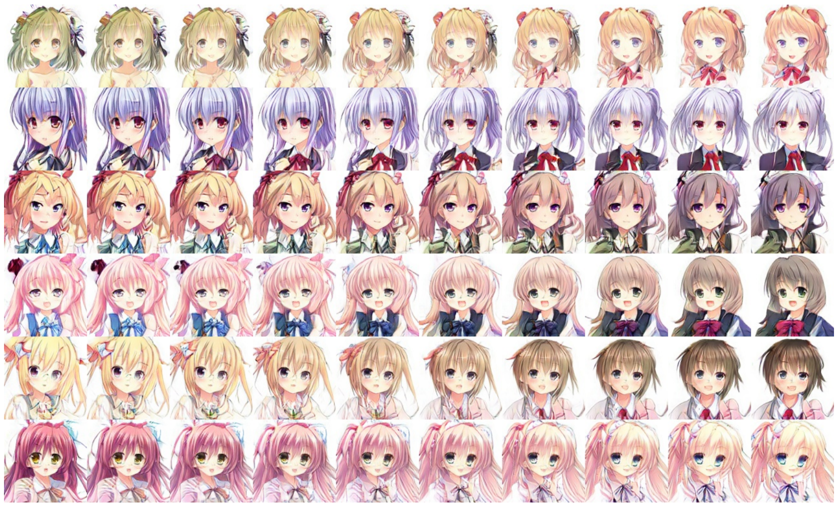


Figure 4.12: Interpolations where samples in the first column and the last columns are randomly generated under different combinations of conditions and samples between them are result of interpolated latent points.

blonde hair 1.00	brown hair 1.00	black hair 1.00	blue hair 0.70	pink hair 0.80	purple hair 0.75	green hair 0.90
red hair 0.95	silver hair 0.85	white hair 0.60	orange hair 0.65	aqua hair 1.00	gray hair 0.35	long hair 1.00
short hair 1.00	twintails 0.60	drill hair 0.20	ponytail 0.45	blush 1.00	smile 0.95	open mouth 0.95
hat 0.15	ribbon 0.85	glasses 0.45	blue eyes 1.00	red eyes 1.00	brown eyes 1.00	green eyes 1.00
purple eyes 0.95	yellow eyes 1.00	pink eyes 0.60	aqua eyes 1.00	black eyes 0.80	orange eyes 0.85	

Table 4.2: Precision of each label.

label, 20 images are drawn from the generator. Then we manually check generated results and judge whether output images behave the fixed attribute we assigned. Table 4.2 shows the evaluation result. From the table, we can see that compared with shape attributes(e.g. “hat”, “glasses”), color attributes are easier to learn. Notice that the boundary between similar colors like “white hair”, “silver hair”, “gray hair ” is not clear enough. Sometimes people may have troubles to classify those confusing colors. This phenomenon leads to low precision scores for those attributes in our test.

Surprisingly, some rare color attributes like “orange eyes”, “aqua hair”, "aqua eyes" have a relative high precisions even though samples containing those attributes are less than 1% in the training dataset. We believe visual concepts related to colors are simple enough for the generator to get well learned with a minimal number of training samples.

In contrast, complex attribute like "hat", "glasses", "drill hair" are worst behaved attributes in our experiments. When conditioned on those labels, generated images are often distorted and difficult to identify. Although there are about 5% training samples assigned with those attributes, the complicated visual concept they implied are far more accessible for the generator to get well learned.

4.5.2 FID Evaluation

One quantitative evaluation method for GAN model is Fréchet Inception Distance (FID) [62]. To calculate the FID, a pre-trained CNN(Inception model) is used to extract vision-relevant features from both real and fake samples. The real feature distribution and the fake feature distribution are approximated with two Gaussian distributions. Then, Fréchet distance(Wasserstein-2 distance) is calculated between two distributions and serve the results as a measurement of the model quality.

The Inception model trained on ImageNet is not suitable for extracting features of anime-style illustrations, since there is no such images in the original training dataset. Here, we replace Inception model with Illustration2vec feature extractor model for better measurement of visual similarities between generated images and real images.

Model	Average FID	MaxFID-MinFID
DCGAN Generator+DRAGAN	5974.96	85.63
Our Model	4607.56	122.96

Table 4.3: FID of our model and baseline model

To evaluate the FID score for our model, we sample 12800 images from real dataset, then generate a fake sample by using the corresponding conditions for each samples real images. After that we feed all images to the Illustation2vec feature extractor and get a 4096-dimension feature vector for each image. FID is calculated between the collection of feature vectors from real samples and that from fake samples.

For each model, we repeat this process for 5 times and measure the average score of 5 FID calculation trails. Table 4.3 shows the result comparing our model with the baseline model. We observe that our model can achieve better FID performance evenly with less weight parameters.

4.5.3 Nearest Image in Training Set

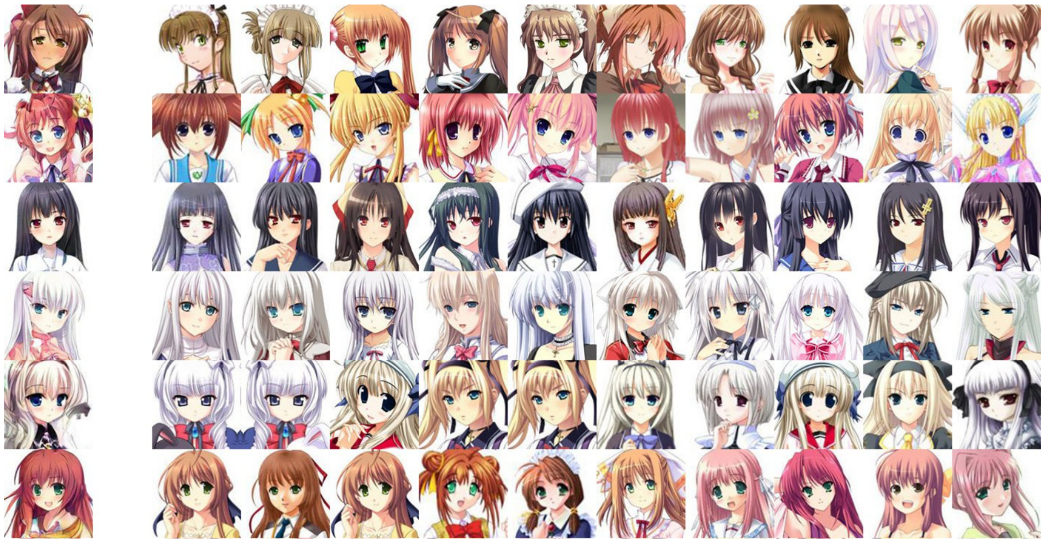


Figure 4.13: Generated images and their nearest image in training set. In each row, on the left we show a randomly generated images, and on the right we show a list of its nearest images in the training set, measured by decreasing L2 distance on features from Illustration2Vec feature extractor. Our model learns to incorporate abstract concepts in terms of attributes (hair style, hair color, etc.) into new, unseen visual features (facial expressions, etc.)

We would like to know whether our model learns to cheat by generating images in the training set. In Figure 4.13 we show randomly generated examples and their nearest images in training set. It can be shown that our model learns to produce images with new visual features unseen in the training set, while incorporating the abstract concepts in terms of attributes.

4.5.4 Evolution of Anime Characters

As an extra experiment, we add the releasing year information as another attribute to the model. Since the main stream of popular anime character styles is continuous evolving, adding year label can help the model catch the prevalent style in each year, as shown in Figure 4.14. Predicting the future of anime character styles is also possible by adjusting the corresponding input value. We made two videos[13] for better demonstrating the result.

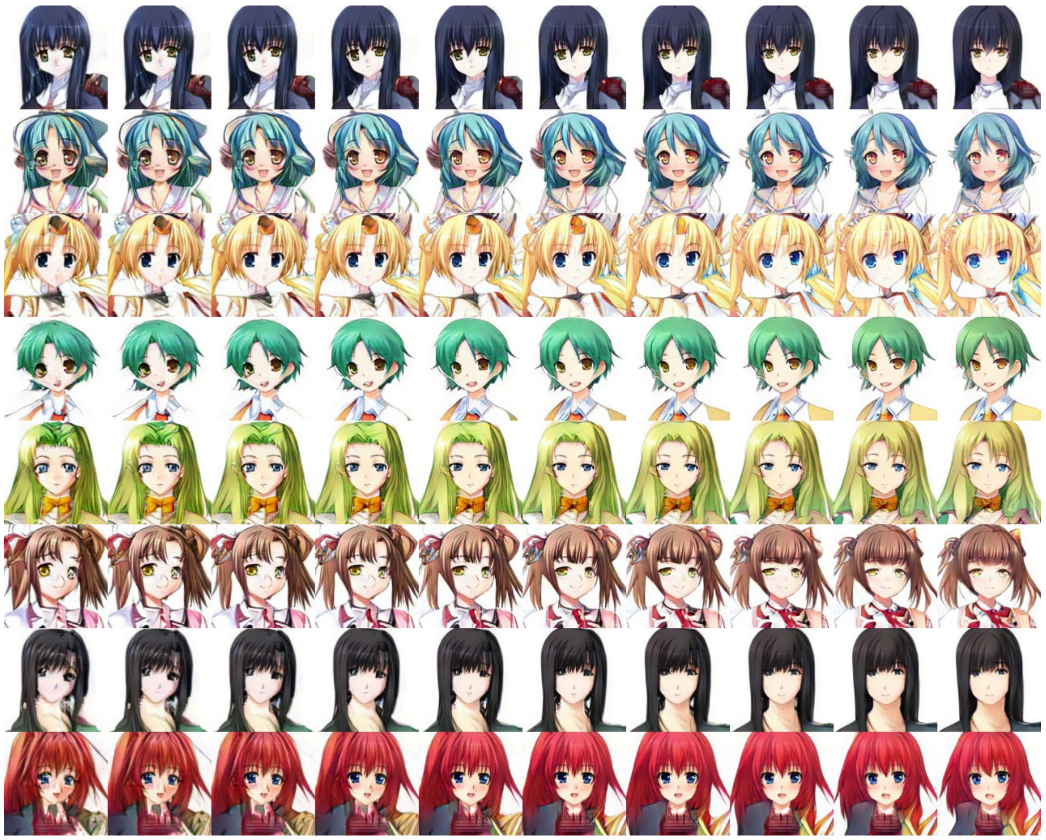


Figure 4.14: Modeling the main stream of popular anime character styles with year label. The leftmost column indicates generated images conditioned on year 2003, and the rightmost column indicates generated images conditioned on year 2017.

[13]https://youtu.be/WR8XnX6W8Bk and https://youtu.be/dClVF_X5PMU

4.6 Public Accessible Interface

We impose WebDNN[14] and convert the trained Chainer model to the WebAssembly based Javascript model. WebGL and WebGPU based computation models are also accessible when clients meet requirements. We observe that the inference procedure can be more than 100 times faster by enabling GPU acceleration. The web application is built with React.js.

Keeping the size of generator model small would be a great benefit when hosting a web browser based deep learning service. This is because user are required to download the model before the computation every time, bigger model results much more downloading time which will affect the user experience. Replacing the DCGAN generator by SRResNet generator can make the model 4x smaller, so the model downloading time can be reduced by a large margin.

We details the running time in Table 4.4. From which we believe that the our support for multiple client-side browsers generally results in acceptable running time and allows smooth user experience, while we also make use of state-of-the-art technology (WebGPU) when it is applicable as a proof-of-concept for next-generation user experience.

Processor	Operation System	Web Browser	Execution Time (s)
I7-6700HQ	macOS Sierra	Chrome 59.0	5.55
I7-6700HQ	macOS Sierra	Safari 10.1	5.60
I5-5250U	macOS Sierra	Chrome 60.0	7.86
I5-5250U	macOS Sierra	Safari 10.1	8.68
I5-5250U	macOS Sierra	Firefox 34	6.01
Intel HD Graphics 6000	macOS Sierra	Safari 11.0(WebGPU)	¡0.10
Intel HD Graphics 6000	macOS Sierra	Chrome 60.0 (WebGL)	0.42
I3-3320	Ubuntu 16.04	Chromium 59.0	53.61
I3-3320	Ubuntu 16.04	Firefox 54.0	4.36
iPhone 7 Plus	iOS 10	Chrome	4.82
iPhone 7 Plus	iOS 10	Safari	3.33
iPhone 6s Plus	iOS 10	Chrome	6.47
iPhone 6s Plus	iOS 10	Safari	6.23
iPhone 6 Plus	iOS 10	Safari	11.55

Table 4.4: Approximate inference time on several different environments.

[14] https://mil-tokyo.github.io/webdnn/

4.7 Improved Training of MakeGirlsMoe

In this section, we briefly describe our effort on improving previous MakeGirlsMoe model by imposing the progressive growing training[77].

4.7.1 Data Preprocessing

As all dataset images have relative low resolution, we first apply the waifu2x[116] super resolution filter to make all dataset images two times larger. Followed by animeface landmark estimation described in Section 4.2.1, we are able to restore the rotation of dataset images and align all anime faces to the sample place.

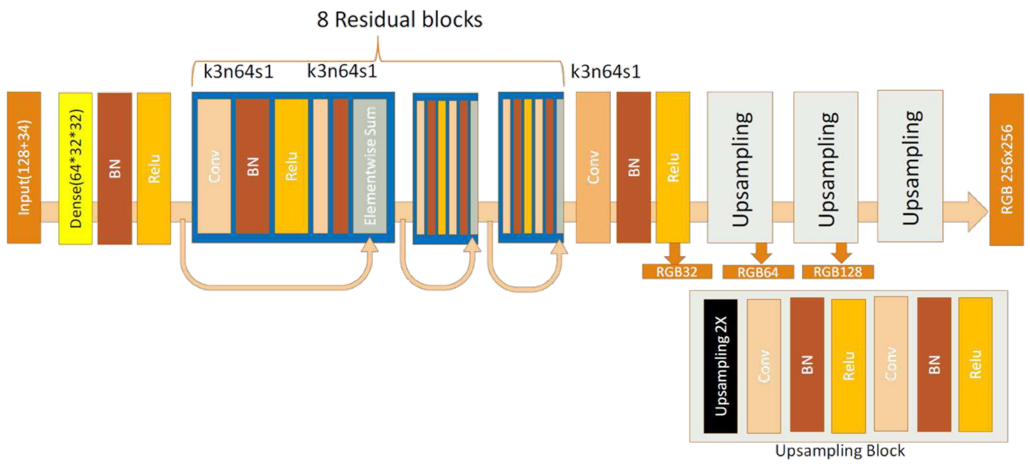


Figure 4.15: Generator architecture for progressive growing training.

4.7.2 Model Description

- Figure 4.15 shows our modified generator architecture. In the training procedure, we first train a 32x32 model, as we believe this is the minimum resolution for clearly identifying facial attributes. Following [77], we then move on to higher resolution models stage by stage.
- An important modification is that we replace the sub-pixel CNN with nearest neighborhood upsampling followed by a convolutional layer in the new model. [1] showed sub-pixel CNN suffers the checkerboard

artifacts caused by random weight initialization, we have confirmed those artifacts in high-resolution GAN training. In [1], they proposed a carefully initialization strategy for tackling the problem, however, the strategy only works for some stable loss functions like mean square loss, which means unacceptable for training a GAN model.

- We also confirm that keeping a running average of generator parameters is critical for a better model(which also called smoothed generator in [77]). This can be viewed as an ensemble of multiple generators.

4.7.3 Generated Results

We show generated results here, the model is already deployed as "Camellia" on our website.

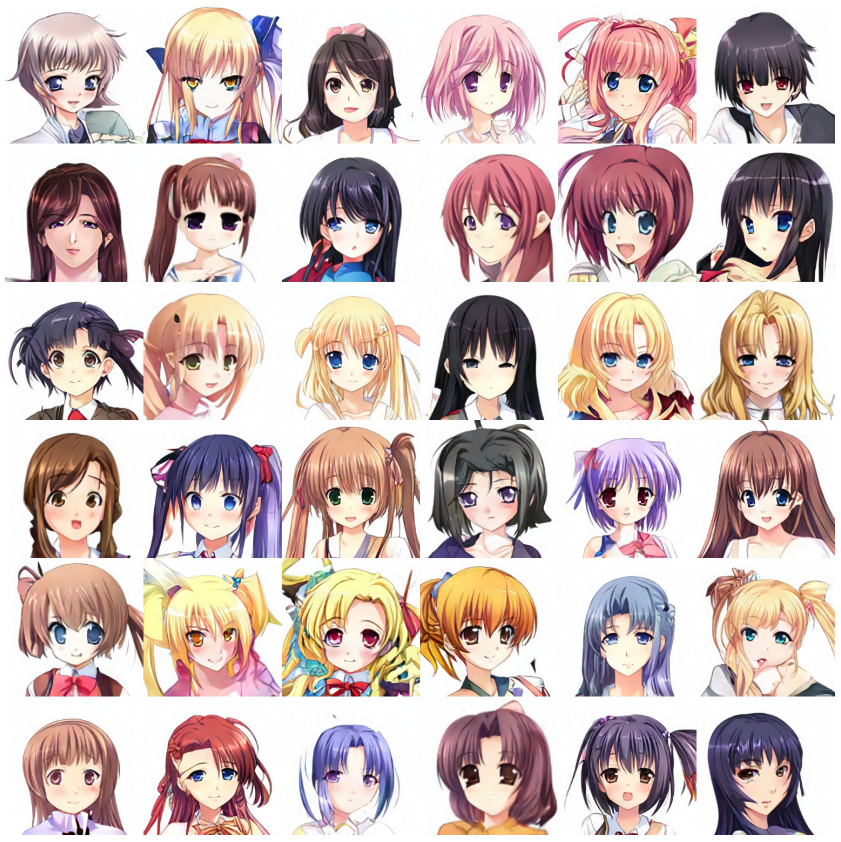


Figure 4.16: Result with random sampled prior.

4.8 Conclusion

This work of anime face generation with high quality opens the door to community-driven creation powered by deep generative methods. Several following up research directions exist.

First and most straightforward, it is interesting to see works going beyond this works limitation of generating face images. For example, the follow-up advance in this direction shows is the feasible generation of full body anime characters [60].

Second, our work shows a certain degree of customization can be feasibly materialized for one user to create artwork according to his/her liking. Therefore it calls for future works to enable finer control-ability of and interactivity between the users, thus improve the quality of robust and coherent character design.

5 Learning to Represent Bilingual Dictionaries [15]

5.1 Introduction

Bilingual word embedding models are used to capture the cross-lingual semantic relatedness of words based on their co-occurrence in parallel or seed-lexicon corpora [25, 56, 104]. By collocating related words in the low-dimensional embedding spaces, these models effectively support the representations of lexical semantics with precise cross-lingual semantic transfer [56]. Therefore, they have been widely used in many cross-lingual NLP tasks including machine translation [42], bilingual document classification [171], knowledge alignment [28] and named entity recognition [53].

While many approaches have been proposed to capture cross-lingual lexical similarity, modeling the correspondence between lexical and sentential semantics across different languages still represents an unresolved challenge. We argue that modeling such cross-lingual and multi-granular correspondence is significant and natural for the following reasons. First, it is highly beneficial to many application scenarios, including cross-lingual semantic search of concepts [148], agents for detecting discourse relations in bilingual dialogue utterances [73], and multilingual text summarization [117], as well as educational applications for foreign language learners. Second, it is natural for a human to learn the meaning of a foreign word by looking up its meaning in the native language. Therefore, learning such correspondence mimics human learning behaviors. Finally, learning word-to-word correspondence can be problematic, since there are words without direct translation in another language. For example, *schadenfreude* in German, which means *a feeling of joy that comes from knowing the troubles of other people*, has no proper English counterpart word. To appropriately learn the representations of such words in bilingual embeddings, we need to capture their meanings based on the definitions as well. However, realizing such a model is a non-trivial task, inasmuch as it requires a comprehensive learning process to effectively compose the semantics of arbitrary-length sentences in one language, and associate that with single words in another language. Consequently, this

[15]The content of this section is taken from:
[115] Muhao Chen* and Yingtao Tian* , Haochen Chen, Kai-Wei Chang, Steven Skiena, and Carlo Zaniolo. Learning to represent bilingual dictionaries. In *Submitted*, 2019

objective also demands high-quality cross-lingual alignment that bridges between single and sequences of words. Such alignment information is generally not available in the parallel and seed-lexicon corpora that are utilized by bilingual word embedding models [56, 86].

To incorporate the representations of bilingual lexical and sentential semantics, we propose an approach by leveraging *bilingual dictionaries* [16]. The proposed approach BilDRL (**Bil**ingual **D**ictionary **R**epresentation **L**earning) seeks to capture *the mapping from word definitions to the corresponding words* in another language. BilDRL first constructs a word embedding space with pre-trained bilingual word embeddings. By utilizing cross-lingual word definitions, a sentence encoder is trained to realize the mapping from literal descriptions to target words in the bilingual word embedding space, for which we investigate with multiple encoding techniques. To enhance cross-lingual learning on limited resources, BilDRL conducts multi-task learning on different directions of language pairs. Moreover, we enforce a joint learning strategy of bilingual word embeddings and the sentence encoder, which seeks to gradually adjust the embedding space to better suit the representation of cross-lingual word definitions.

To show the applicability of BilDRL, we conduct experimental evaluation on two useful cross-lingual tasks (see Fig. 5.1). (i) *Cross-lingual reverse dictionary retrieval* seeks to retrieve words or concepts given descriptions in another language. This task is useful to help users find foreign words based on the notions or descriptions, and is especially beneficial to users such as translators, foreigner language learners and technical writers using non-native languages. We show that BilDRL achieves promising results on this task, while bilingual multi-task learning and joint learning dramatically enhance the performance. (ii) *Bilingual paraphrase identification* asks whether two sentences in different languages essentially express the same meaning, which is critical to question answering or dialogue systems that apprehend multilingual utterances [12]. This task is challenging, as it requires a model to comprehend cross-lingual paraphrases that are inconsistent in grammar, content details and word orders. BilDRL maps sentences to the lexicon embedding space. This process reduces the problem to evaluate the similarity of lexicon embeddings, which can be easily solved by a simple classifier.

[16]We refer the term *dictionary* to its regular meaning, i.e. lexicographic definitions of words. Note that this is different from many recent papers on bilingual settings that refer dictionaries to seed lexicons, e.g. one-to-one word mapping.

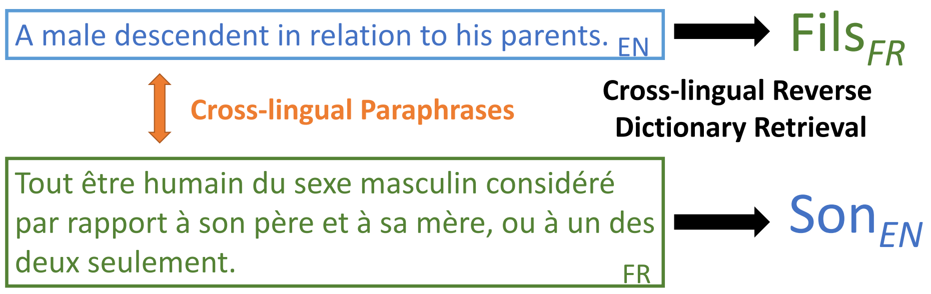

A male descendent in relation to his parents. EN
Fils FR
Cross-lingual Reverse Dictionary Retrieval
Cross-lingual Paraphrases
Tout être humain du sexe masculin considéré par rapport à son père et à sa mère, ou à un des deux seulement. FR
Son EN

Figure 5.1: An example illustrating the two cross-lingual tasks. The *cross-lingual reverse dictionary retrieval* finds cross-lingual target words based on descriptions. In terms of *cross-lingual paraphrases*, the French sentence (which means *any male being considered in relation to his father and mother, or only one of them*) essentially describes the same meaning as the English sentence, but has much more content details.

BilDRL performs well with even a small amount of data, and significantly outperforms previous approaches.

5.2 Related Work

In this section, We discuss two lines of work that are relevant to our topic.
Bilingual word embeddings. Recently, various approaches have been proposed for training bilingual word embeddings. These approaches span in two families: off-line mappings and joint training.

The off-line mapping-based approach fixes the structures of pre-trained monolingual word embeddings, and induces bilingual projections based on seed-lexicon alignment [108]. Some variants of this approach improve the quality of bilingual projections by adding constraints such as orthogonality of transforms, normalization and mean centering of embeddings [164, 9]. Others adopt canonical correlation analysis to map separated monolingual embeddings to a shared embedding space [52, 103].

Unlike off-line mappings, joint training models simultaneously update word embeddings and cross-lingual alignment. In doing so, such approaches generally capture more precise cross-lingual semantic transfer [150]. While few of such models still maintain separated embedding spaces for each language [67], more recent ones obtain a unified space for both languages. The cross-lingual semantic transfer by these models is captured from parallel corpora with sentential or document-level alignment, using techniques such as bilingual bag-of-words distances (BilBOWA) [56], Skip-Gram [35] and sparse tensor factorization [153].

Neural sentence modeling. Neural sentence models seek to characterize the phrasal or sentential semantics from word sequences. They often adopt encoding techniques such as recurrent neural encoders (RNN) [80], convolutional neural encoders (CNN) [27], and attentive encoders [129] to represent the composed semantics of a sentence as an embedding vector. Recent works have focused on apprehending pairwise correspondence of sentential semantics by adopting multiple neural sentence models in one learning architecture, including Siamese models for detecting discourse relations of paraphrases or text entailment [133], sequence-to-sequence models for tasks like style transfer [134] and text summarization [32]. While our work is related to corresponding works of neural machine translation (NMT) [11, 161], our setting has major differences from NMT in the following two perspectives: (i) NMT has to bridge between corpora of the same granularity, unlike BilDRL that captures the multi-granular correspondence of semantics across different modalities; (ii) NMT relies on training an encoder-decoder architecture, while BilDRL employs joint learning of two representation models, i.e. a dictionary-based sentence encoder and a word embedding model.

On the other hand, fewer efforts have been put to characterizing the associations between sentential and lexical semantics. [64] and [72] learn off-line mappings between monolingual descriptions and lexicons to capture such associations. [46] adopt a similar approach to capture emojis based on descriptions. At the best of our knowledge, there has been no previous approach that learn to discover the correspondence of sentential and lexical semantics in a multilingual scenario. This is exactly the focus of our work, in which the proposed strategies of multi-task and joint learning are critical to the corresponding learning process under limited resources. Utilizing such correspondence, our approach also sheds light on addressing discourse relation detection in a multilingual scenario.

5.3 Modeling Bilingual Dictionaries

We hereby begin our modeling with the formalization of bilingual dictionaries. We use $\mathcal{L}$ to denote the set of languages. For a language $l \in \mathcal{L}$, V_l denotes its vocabulary, where for each word $w \in V_l$, bold-faced $\mathbf{w} \in \mathbb{R}^k$

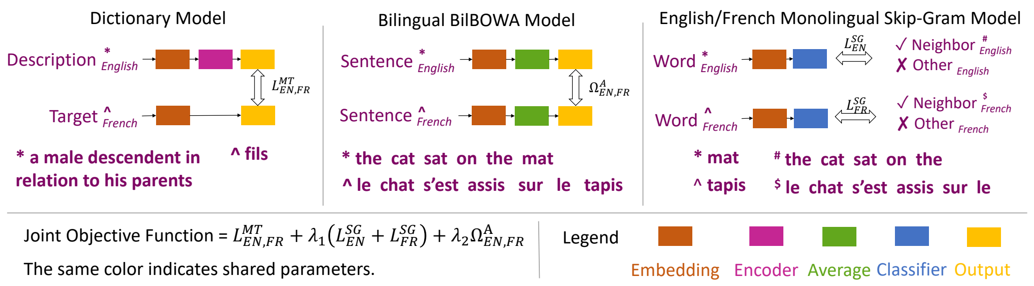


Figure 5.2: Joint learning architecture of BilDRL.

denotes its embedding vector. A l_i-l_j bilingual dictionary $D(l_i, l_j)$ (or simply D_{ij}) contains dictionary entries $(w^i, S^j_w) \in D_{ij}$, in which $w^i \in V_{l_i}$, and $S^j_w = w^j_1 \dots w^j_n$ $(w^j_\cdot \in V_{l_j})$ is a cross-lingual word definition that describes the word w^i with a sequence of words in language l_j. For example, a French-English dictionary $D(\mathrm{Fr}, \mathrm{En})$ could include a French word *appétite* accompanied by its English definition *desire for, or relish of food or drink*. Note that, for a word w^i, multiple definitions in l_j may coexist.

BilDRL is constructed and improved through three stages, as depicted in Fig. 5.2. A sentence encoder is first used to learn from a bilingual dictionary the association between words and definitions. Then in a pre-trained bilingual word embedding space, multi-task learning is conducted on both directions of a language pair. Lastly, joint learning with word embeddings is enforced to simultaneously adjust the embedding space during the training of the dictionary model, which further enhances the cross-lingual learning process.

5.3.1 Encoders for Bilingual Dictionaries

BilDRL models a dictionary using a neural sentence encoder $E(S)$, which composes the meaning of the sentence into a latent vector representation. We hereby introduce this model component, which is designed to be a GRU encoder with self-attention. Besides that, we also investigate other widely-used neural sentence modeling techniques.

Attentive GRU Encoder The GRU encoder is a computationally efficient alternative of the LSTM [31]. Each unit consists of a reset gate $\mathbf{r}_t$ and an update gate $\mathbf{z}_t$ to track the state of the sequence. Given the vector representation $\mathbf{w}_t$ of an incoming item w_t, GRU updates the hidden state $\mathbf{h}_t^{(1)}$ as a linear combination of the previous state $\mathbf{h}_{t-1}^{(1)}$ and the candidate state $\tilde{\mathbf{h}}_t^{(1)}$ of the new item w_t as below.

$$\mathbf{h}_t^{(1)} = \mathbf{z}_t \odot \tilde{\mathbf{h}}_t^{(1)} + (1 - \mathbf{z}_t) \odot \mathbf{h}_{t-1}^{(1)}$$

The update gate $\mathbf{z}_t$ balances between the information of the previous sequence and the new item, where $\mathbf{M}_z$ and $\mathbf{N}_z$ are two weight matrices, $\mathbf{b}_z$ is a bias vector, and σ is the sigmoid function.

$$\mathbf{z}_t = \sigma\left(\mathbf{M}_z \mathbf{x}_t + \mathbf{N}_z \mathbf{h}_{t-1}^{(1)} + \mathbf{b}_z\right)$$

The candidate state $\tilde{\mathbf{h}}_t^{(1)}$ is calculated similarly to those in a traditional recurrent unit as below.

$$\tilde{\mathbf{h}}_t^{(1)} = \tanh\left(\mathbf{M}_s \mathbf{w}_t + \mathbf{r}_t \odot (\mathbf{N}_s \mathbf{h}_{t-1}^{(1)}) + \mathbf{b}_s\right)$$

The reset gate $\mathbf{r}_t$ thereof controls how much information of the past sequence should contribute to the candidate state:

$$\mathbf{r}_t = \sigma\left(\mathbf{M}_r \mathbf{w}_t + \mathbf{N}_r \mathbf{h}_{t-1}^{(1)} + \mathbf{b}_r\right)$$

The above defines a GRU layer which outputs a sequence of latent vectors given an input sequence S. While a GRU encoder can stack multiple GRU layers, without an attention mechanism, the last hidden state $\mathbf{h}_S^{(1)}$ of the last layer is used to represent the overall meaning of the encoded sentence.

Self-attention. The self-attention mechanism [34] seeks to highlight the important units in an input sentence when capturing its overall meaning, which is calculated as below:

$$\begin{aligned}
\mathbf{u}_t &= \tanh\left(\mathbf{M}_a \mathbf{h}_t^{(1)} + \mathbf{b}_a\right) \\
a_t &= \frac{\exp\left(\mathbf{u}_t^\top \mathbf{u}_S\right)}{\sum_{w_m \in S} \exp\left(\mathbf{u}_m^\top \mathbf{u}_S\right)} \\
\mathbf{h}_t^{(2)} &= |S| a_t \mathbf{u}_t
\end{aligned}$$

$\mathbf{u}_t$ thereof is the intermediary representation of GRU output $\mathbf{h}_t^{(1)}$, and $\mathbf{u}_S = \tanh(\mathbf{M}_a \mathbf{h}_S^{(1)} + \mathbf{b}_a)$ is that of the last GRU output $\mathbf{h}_S^{(1)}$, which can be seen as a high-level representation of the entire input sequence. By measuring the similarity of each $\mathbf{u}_t$ with $\mathbf{u}_S$, the normalized attention weight a_t is produced through a softmax function, which highlights an input that contributes more significantly to the overall meaning. Note that a scalar $|S|$ is multiplied along with a_t to $\mathbf{u}_t$ to obtain the weighted representation $\mathbf{h}_t^{(2)}$, so as to keep $\mathbf{h}_t^{(2)}$ from losing the original scale of $\mathbf{h}_t^{(1)}$. The sentence encoding is calculated as the average of the last attention layer $E^{(1)}(S) = \frac{1}{|S|}\sum_{t=1}^{|S|} a_t\mathbf{h}_t^{(2)}$.

Other Encoders We also experiment with other widely used neural sentence modeling techniques, which are however outperformed by the attentive GRU encoder in our tasks. These techniques include the vanilla GRU, CNN [76], and linear bag-of-words (BOW) [64]. We briefly introduce the later two techniques in the following.
Convolutional Encoder. A convolutional encoder applies a kernel $\mathbf{M}_c \in \mathbb{R}^{h\times k}$ to produce the latent representation $\mathbf{h}_t^{(3)} = \tanh(\mathbf{M}_c\mathbf{w}_{t:t+h-1} + \mathbf{b}_c)$ from each h-gram of the input vector sequence $\mathbf{w}_{t:t+h-1}$, for which h is the kernel size and $\mathbf{b}_c$ is a bias vector. A sequence of latent vectors $\mathbf{H}^{(3)} = [\mathbf{h}_1^{(3)}, \mathbf{h}_2^{(3)}, ..., \mathbf{h}_{|S|-h+1}^{(3)}]$ is produced from the input, where each latent vector leverages the significant local semantic features from each h-gram. Following convention [98], we apply dynamic max-pooling to extract robust features from the convolution outputs, and use the mean-pooling results of the last layer to represent the sentential semantics.
Linear bag-of-words. The much simpler BOW encoder [72, 64] is realized by the sum of projected word embeddings of the input sentence, i.e. $E^{(2)}(S) = \sum_{t=1}^{|S|}\mathbf{M}_b\mathbf{w}_t$.

5.3.2 Basic Learning Objective

The objective of learning the dictionary model is to map the encodings of cross-lingual word definitions to the target word embeddings. This is realized by minimizing the following L_2 loss,

$$L_{ij}^{\mathrm{ST}} = \frac{1}{|D_{ij}|}\sum_{(w^i, S_w^j)\in D_{ij}} \left\|E_{ij}(S_w^j) - \mathbf{w}^i\right\|_2^2$$

in which E_{ij} is the dictionary model that maps from descriptions in l_j to words in l_i.

The above defines the basic model variants of BilDRL that learns on a single dictionary. For word representations in the learning process, BilDRL initializes the embedding space using pre-trained word embeddings. Note that, without adopting the joint learning strategy in Section 5.3.4, the learning process does not update word embeddings that are used to represent the definitions and target words. While other forms of loss such as cosine proximity [64] and hinge loss [72] may also be used in the learning process, we find that L_2 loss consistently leads to better performance in our experiments.

5.3.3 Bilingual Multi-task Learning

In cases where entries in a bilingual dictionary are not amply provided, learning the above bilingual dictionary on one ordered language pair may fall short in insufficiency of alignment information. One practical solution is to conduct a bilingual multi-task learning process. In detail, given a language pair (l_i, l_j), we learn the dictionary model E_{ij} on both dictionaries D_{ij} and D_{ji} with shared parameters. Correspondingly, we rewrite the previous learning objective function as below, in which $D = D_{ij} \cup D_{ji}$.

$$L_{ij}^{\mathrm{MT}} = \frac{1}{|D|} \sum_{(w,S_w)\in D} \|E_{ij}(S_w) - \mathbf{w}\|_2^2$$

This strategy non-trivially requests the same dictionary model to represent semantic transfer in two directions of the language pair. To fulfill such a request, we initialize the embedding space using the bilingual BilBOWA embeddings trained on parallel corpora, which provides a unified embedding space that resolves both monolingual and cross-lingual semantic relatedness of words. In practice, we find that this simple multi-task strategy brings significant improvement to our cross-lingual tasks. Note that, besides BilBOWA, other jointly trained bilingual word embeddings [35, 153] may also be used to support this strategy, for which we leave the comparison to future work.

5.3.4 Joint Learning

While above learning strategies of BilDRL are based on a fixed embedding space, we lastly propose a joint learning strategy. During the training process, this strategy simultaneously updates the embedding space based on both the dictionary model and the bilingual word embedding model. The learning is through asynchronous minimization of the following joint objective function,

$$J = L_{ij}^{\mathrm{MT}} + \lambda_1(L_i^{\mathrm{SG}} + L_j^{\mathrm{SG}}) + \lambda_2\Omega_{ij}^{\mathrm{A}}$$

where λ_1 and λ_2 are two positive coefficients. L_i^{SG} and L_j^{SG} are the original Skip-Gram losses [109] employed by BilBOWA to separately obtain word embeddings on monolingual corpora of languages l_i and l_j. Ω_{ij}^{A} is the alignment loss to minimize the bag-of-words distances for aligned sentence pairs (S^i, S^j) from the parallel corpora C_{ij}, which is termed as below.

$$\Omega_{ij}^{\mathrm{A}} = \frac{1}{|C_{ij}|}\sum_{(S^i,S^j)\in C_{ij}} d_{\mathrm{S}}(S^i, S^j)$$

$$d_{\mathrm{S}}(S^i, S^j) = \left\| \frac{1}{|S^i|}\sum_{w_m^i\in S^i}\mathbf{w}_m^i - \frac{1}{|S^j|}\sum_{w_n^j\in S^j}\mathbf{w}_n^j \right\|_2^2$$

The joint learning process adapts the embedding space to better suit the dictionary model, which is shown to further enhance the cross-lingual learning of BilDRL.

5.3.5 Training

To initialize the embedding space, we pre-trained BilBOWA on the parallel corpora Europarl v7 [83] and monolingual corpora of tokenized Wikipedia dump [2]. For models without joint learning, we use AMSGrad [123] to optimize the parameters. Each model without bilingual multi-task learning thereof, is trained on batched samples from each individual dictionary. Multi-task learning models are trained on batched samples from two dictionaries. Within each batch, entries of different directions of languages can be mixed together. For joint learning, we follow previous works [56, 112] to conduct an efficient multi-threaded asynchronous training [111] of AMSGrad. In detail, after initializing the embedding space based on pre-trained BilBOWA, parameter updating based on the four components of J occurs across four worker threads. Two monolingual threads select batches of monolingual contexts from the Wikipedia dump of two languages for Skip-Gram, one alignment thread randomly samples parallel sentences from Europarl, and one dictionary thread extracts batched samples of entries for a bilingual multi-task dictionary model. Each thread makes a batched update to model parameters asynchronously for each component of J. The asynchronous training of all threads keeps going until the dictionary thread finishes its epochs.

5.4 Experiments

In this section, we present the experiments on two cross-lingual tasks: the cross-lingual reverse dictionary retrieval task and the bilingual paraphrase identification task.

Datasets. The experiment of cross-lingual reverse dictionary retrieval is conducted on a trilingual dataset *Wikt3l*. This dataset is extracted from Wiktionary[17], which is one of the largest freely available multilingual dictionary resources on the Web. Wikt3l contains dictionary entries of language pairs (English, French) and (English, Spanish), which form En-Fr, Fr-En, En-Es and Es-En dictionaries on four bridges of languages in total. Two types of bilingual dictionary entries are extracted from Wiktionary: (i) cross-lingual definitions provided under the *Translations* sections of Wiktionary pages; (ii) monolingual definitions for words that are linked to a cross-lingual counterpart with a inter-language link[18] of Wiktionary. We exclude all the definitions of stop words in constructing the dataset, and list the statistics in Table 5.1.

Since existing datasets for paraphrase identification are merely monolingual, we contribute with another dataset *WBP3l* for cross-lingual sentential paraphrase identification. This dataset contains 6,000 pairs of bilingual sentence pairs respectively for En-Fr and En-Es settings. Within each bilingual setting, 3,000 positive cases are formed as pairs of descriptions aligned by inter-language links, which exclude the word descriptions in Wikt3l for training BilDRL. To generate negative examples, given a source word, we first find its 15 nearest neighbors in the embedding space. Then we randomly pick one word from these neighbors and pair its cross-lingual definition with the English definition of the source word to create a negative case. This process ensures that each negative case is endowed with limited dissimilarity of sentence meanings, which makes the decision more challenging. For each language setting, we randomly select 70% for training, 5% for validation, and the rest 25% for testing. Note that each language setting of this dataset

[17]https://www.wiktionary.org/

[18]An inter-language link matches the entries of counterpart words between language versions of Wiktionary.

Dictionary	En-Fr	Fr-En	En-Es	Es-En
#Target words	15,666	16,857	8,004	16,986
#Definitions	50,412	58,808	20,930	56,610

Table 5.1: Statistics of the bilingual dictionary dataset Wikt3l.

Positive Examples
En: *Being remote in space.*   **Fr**: *Se trouvant à une grande distance.*
En: *The interdisciplinary science that applies theories and methods of the physical sciences to questions of biology.*   **Es**: *Ciencia que emplea y desarrolla las teoras y métodos de la física en la investigación de los sistemas biolgicos.*
Negative Examples
En: *A person who secedes or supports secession from a political union.*   **Fr**: *Contrôle politique exercé par une grande puissance sur une contre inféodée.*
En: *The fear of closed, tight places.*   **Es**: *Pérdida o disminución considerables de la memoria.*

Table 5.2: Examples of bilingual paraphrases from WBP3l.

thereof, matches with the quantity and partitioning of sentence pairs in the widely-used Microsoft Research Paraphrase Corpus benchmark for monolingual paraphrase identification [167, 39]. Several examples from the dataset are shown in Table 5.2.

5.4.1 Cross-lingual Reverse Dictionary Retrieval

The objective of this task is to enable cross-lingual semantic retrieval of words based on descriptions. Besides comparing variants of BilDRL that adopt different sentence encoders and learning strategies, we also compare with the monolingual retrieval approach proposed by [64]. Instead of directly associating cross-lingual word definitions, this approach learns definition-to-word mappings in a monolingual scenario. When it applies to the multilingual

Languages	En-Fr			Fr-En			En-Es			Es-En		
Metric	P@1	P@10	*MRR*	P@1	P@10	*MRR*	P@1	P@10	*MRR*	P@1	P@10	*MRR*
BOW	0.8	3.4	0.011	0.4	2.2	0.006	0.4	2.4	0.007	0.4	2.6	0.007
CNN	6.0	12.4	0.070	6.4	14.8	0.072	3.8	7.2	0.045	7.0	16.8	0.088
GRU	35.6	46.0	0.380	38.8	49.8	0.410	47.8	59.0	0.496	57.6	67.2	0.604
ATT	38.8	47.4	0.411	39.8	50.2	0.425	51.6	59.2	0.534	60.4	68.4	0.629
GRU-mono	21.8	33.2	0.242	27.8	37.0	0.297	34.4	41.2	0.358	36.8	47.2	0.392
ATT-mono	22.8	33.6	0.249	27.4	39.0	0.298	34.6	42.2	0.358	39.4	48.6	0.414
GRU-MTL	43.4	49.2	0.452	44.4	52.8	0.467	50.4	60.0	0.530	63.6	71.8	0.659
ATT-MTL	46.8	56.6	0.487	47.6	56.6	0.497	55.8	62.2	0.575	66.4	75.0	0.687
ATT-joint	**63.6**	**69.4**	**0.654**	**68.2**	**75.4**	**0.706**	**69.0**	**72.8**	**0.704**	**78.6**	**83.4**	**0.803**

Table 5.3: Cross-lingual reverse dictionary retrieval results by BilDRL variants. We report P@1, P@10, and *MRR* on four groups of models: (i) basic dictionary models that adopt four different encoding techniques (BOW, CNN, GRU and ATT); (ii) models with the two best encoding techniques that enforce the monolingual retrieval approach by [64] (GRU-mono and ATT-mono); (iii) models adopting bilingual multi-task learning (GRU-MTL and ATT-MTL); (iv) joint learning that employs the best dictionary model of ATT-MTL (ATT-joint).

setting, given a word definition, it first retrieve the corresponding word in the source language. Then, it looks up for semantically related words in the target language using bilingual word embeddings.

Evaluation Protocol. Before training the models, we randomly select 500 defined words from each dictionary respectively as test cases, and exclude all the definitions of these words from the rest training data. Each of the basic BilDRL variants are trained on one bilingual dictionary. The monolingual retrieval models are trained to fit the target words in the original languages of the word definitions, which are also provided in Wiktionary. BilDRL variants with multi-task or joint learning use both dictionaries on the same language pair. In the test phase, for each test case $(w^i, S_w^j) \in D_{ij}$, the prediction by each model is to perform a kNN search from the corresponding definition encoding $E_{ij}(S_w^j)$, and record the rank of w^i within the vocabulary of l_i. We limit the vocabularies to all the words that appear in the Wikt3l dataset, which involve around 45k English words, 44k French words and 36k Spanish words. We aggregate three metrics on test cases: the accuracy P@1 (%), the proportion of ranks no larger than 10 $P@10$ (%), and mean reciprocal rank *MRR*.

We pre-train BilBOWA based on the original configuration in [56] and obtain 50-dimensional initialization of bilingual word embedding spaces respectively for the English-French and English-Spanish settings. For CNN, GRU, and attentive GRU (ATT) encoders, we stack five of each corresponding encoding layers with hidden-sizes of 200, and two affine layers are applied to the final output for dimension reduction. This encoder architecture consistently represents the best performance through our tuning. Through comprehensive hyperparameter tuning, we fix the learning rate α to 0.0005, the exponential decay rates of AMSGrad β_1 and β_2 to 0.9 and 0.999, coefficients λ_1 and λ_2 to both 0.1, and batch size to 64. Kernel-size and pooling-size are both set to 2 for CNN. Word definitions are zero-padded (short ones) or truncated (long ones) to the sequence length of 15, since most definitions (over 92%) are within 15 words in the dataset. Training is limited to 1,000 epochs for all models as well as the dictionary thread of asynchronous joint learning, in which all models are able to converge.

Results. Results are reported in Table 5.3 in four groups. The first group compares four different encoding techniques for the basic dictionary models. GRU thereof consistently outperforms CNN and BOW, since the latter two fail to capture the important sequential information for descriptions. ATT that weighs among the hidden states has notable improvements over GRU. While we equip the two better encoding techniques with the monolingual retrieval approach (GRU-mono and ATT-mono), we find that the way of learning the dictionary models towards monolingual targets and retrieving cross-lingual related words incurs more impreciseness to the task. For models of the third group that conduct multi-task learning in two directions of a language pair, the results show significant enhancement of performance in both directions. For the final group of results, we incorporate the best variant of multi-task models into the joint learning architecture, which leads to compelling improvement of the task on all settings. This demonstrates that properly adapting the word embeddings in joint with the bilingual dictionary model efficaciously constructs the embedding space that suits better the representation of both bilingual lexical and sentential semantics.

In general, this experiment has identified the proper encoding techniques of the dictionary model. The proposed strategies of multi-task and joint learning effectively contribute to the precise characterization of the cross-lingual correspondence of lexical and sentential semantics, which have led to very promising capability of cross-lingual reverse dictionary retrieval.

5.4.2 Bilingual Paraphrase Identification

The bilingual paraphrase identification problem is a binary classification task with the goal to decide whether two sentences in different languages express the same meanings[19]. BilDRL provides an effective solution by transferring sentential meanings to lexicon-level representations and learning a simple classifier. We evaluate three variants of BilDRL on this task using WBP3l: the multi-task BilDRL with GRU encoders (BilDRL-GRU-MTL), the multi-task BilDRL with attentive GRU encoders (BilDRL-ATT-MTL), and the joint learning BilDRL with with attentive GRU encoders (BilDRL-ATT-joint). We compare against several baselines of neural sentence pair models that are proposed for monolingual paraphrase identification. These models include Siamese structures of CNN (BiCNN) [166], RNN (BiGRU) [114], attentive CNN (ABCNN) [167], attentive GRU (BiATT) [129], and linear BOW (BiBOW). To support the reasoning of cross-lingual semantics, we provide these baselines with the same BilBOWA embeddings.

Evaluation protocol. BilDRL transfers each sentence into a vector in the word embedding space. Then, for each sentence pair in the train set, a Multi-layer Perceptron (MLP) with a binary softmax loss is trained on the subtraction of two vectors as a downstream classifier. Baseline models are trained end-to-end, each of which directly uses a parallel pair of encoders with shared parameters and an MLP that is stacked to the subtraction of two sentence vectors. Note that some works use concatenation [166] or Manhattan distance [114] of sentence vectors instead of their subtraction [73], which we find to be less effective on small amount of data.

We apply the configurations of the sentence encoders from the last experiment to corresponding baselines, so as to show the performance under controlled variables. Training of a classifier is terminated by early-stopping based on the validation set. Following convention [65, 167], we report the accuracy and F1 scores.

Results. This task is challenging due to the heterogeneity of cross-lingual

[19]Paraphrases have similar meanings, but can differ a lot in content details and have inconsistent word orders. Hence, they are essentially different from simple translations of sentences. We have found that even the well-recognized Google NMT frequently causes distortions to short sentence meanings, and lead to the results that are close to random guess from the baseline classifiers after translation.

Languages	En&Fr		En&Es	
Metrics	*Acc.*	*F1*	*Acc.*	*F1*
BiBOW	54.93	0.622	56.27	0.623
BiCNN	54.33	0.625	53.80	0.611
ABCNN	56.73	0.644	58.83	0.655
BiGRU	60.81	0.697	60.53	0.692
BiATT	61.47	0.699	61.27	0.689
BilDRL-GRU-MTL	64.80	0.732	63.33	0.722
BilDRL-ATT-MTL	65.27	0.735	66.07	0.735
BilDRL-ATT-joint	**68.53**	**0.785**	**67.13**	**0.759**

Table 5.4: Accuracy and F1-scores of bilingual paraphrase identification. For BilDRL, the results by three model variants are reported: BilDRL-GRU-MTL and BilDRL-ATT-MTL are models with bilingual multi-task learning, and BilDRL-ATT-joint is the best ATT-based dictionary model variant deployed with both multi-task and joint learning.

paraphrases and limitedness of learning resources. where BiATT consistently outperforms the others, merely reaches slightly over 60% of accuracy on both En-Fr and En-Es settings. We believe that it comes down to the fact that sentences of different languages are often drastically heterogenous in both lexical semantics and the sentence grammar that governs the composition of lexicons. Hence, it is not surprising that previous neural sentence pair models, which capture the semantic relation of bilingual sentences directly from all participating lexicons, fall short at the multilingual task. BilDRL, however, effectively leverages the correspondence of lexical and sentential semantics to simplify the task to an easier entailment task in the lexicon space, for which the multi-task learning BilDRL-ATT-MTL outperforms the best baseline respectively by 3.80% and 4.80% of accuracy in both language settings, while BilDRL-ATT-joint, employing the joint learning, further improves the task by another satisfying 3.26% and 1.06% of accuracy. Both also show notable increment in F1.

5.5 Conclusion

In this paper, we propose a neural embedding model BilDRL that captures the correspondence of cross-lingual lexical and sentential semantics. We experiment with multiple forms of neural models and identify the best technique. The two learning strategies, bilingual multi-task learning and joint learning, are effective at enhancing the cross-lingual learning with limited resources, and also achieve promising performance on cross-lingual reverse dictionary retrieval and bilingual paraphrase identification tasks by associating lexical and sentential semantics.

We identify several important directions of future works to explore given the contribution presented and the limitation of scope assumed in this work. One direction is to explore whether the lexicon-sentence alignment can improve bilingual word embeddings. Bilingual embeddings[175, 152] and sentence embeddings [114, 160] have been well-studied separately, Therefore the observation of this work that putting them in the same space helps specific tasks may lead to interesting investigation on whether it can make bilingual embeddings themselves a better one. Another important direction concerns applying BilDRL to bilingual question answering and semantic search systems. Again we observe that question answering and semantic search in the multilingual setting are built either separately on all involved languages or ad-/post-hocing a separate, general-purpose translation system. Either of them does not fully leverage the benefit of bilingual task-specific data, so we expect this to be another place where our work may provide insight.

6 Latent Translation: Crossing Modalities by Bridging Generative Models [20]

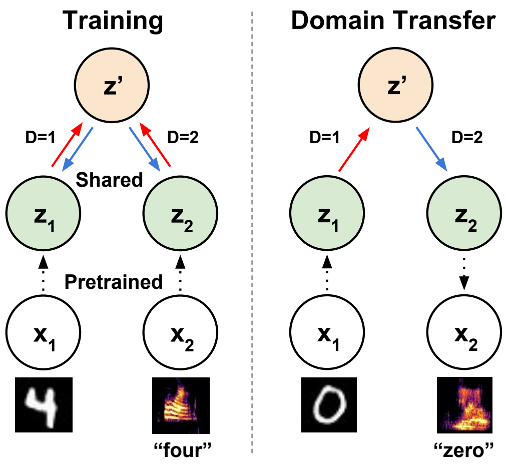


Figure 6.1: Latent translation with a shared autoencoder. Pretrained generative models provide embeddings (z_1, z_2) for data in two different domains (x_1, x_2), here shown as written digits and (spectrograms of) spoken digits. A shared autoencoder creates joint embeddings (z') which are encouraged to overlap by a sliced-wasserstein distance and semantically structured by a linear classifier. The autoencoder is trained with an additional one-hot domain label (D), and domain transfer occurs by encoding with one domain label and decoding with the other. More details are available in Section 6.3.

6.1 Introduction

Modularity enables general and efficient problem solving by recombining predefined components in new ways. As such, modular design is essential to such core pursuits as proving theorems, designing algorithms, and software engineering. Despite the many advances of deep learning, including impressive generative models such as Variational Autoencoders (VAEs) and Generative

[20]The content of this section is taken from:
[147] Yingtao Tian and Jesse Engel. Latent translation: Crossing modalities by bridging generative models. In *Submitted*, 2019

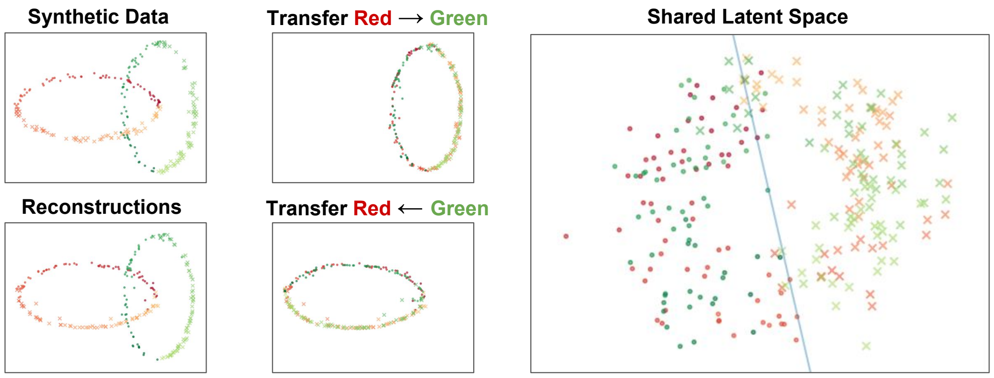


Figure 6.2: Synthetic problem demonstrating latent translation (best viewed in color). Synthetic datasets are created to represent pretrained embeddings for two data domains (red and green ellipses, 2-dimensional). A small "bridging" autoencoder (as in Figure 6.1) is trained to reconstruct data from both domains. The shared latent space (also 2-dimensional) has domain overlap because of the SWD penalty, and class separation due to the linear classifier (decision boundary shown). This enables bidirectional domain transfer that preserves local structure (shown by the color gradient of datapoints) and class separation (learning to rotate rather than shift and squash).

Adversarial Networks (GANs), end-to-end optimization inhibits modular design by providing no straight-forward way to combine predefined modules. While performance benefits still exist for scaling individual models, such as the recent BigGAN model [23], it becomes increasingly infeasible to retrain such large architectures for every new use case. By analogy, this would be equivalent to having to rewrite an assembly compiler every time one wanted to write a new web application.

Transfer learning and fine-tuning are common methods to reuse individual pretrained modules for new tasks [41, 101], but there is still no clear method to get the combinatorial benefits of integrating multiple modules. Here, we explore one such approach by bridging pretrained latent generative models to perform cross-modal domain transfer.

For cross-modal domain transfer, we seek to train a model capable of transferring instances from a source domain (x_1) to a target domain (x_2), such that local variations in source domain are transferred to local variations in the target domain. We refer to this property as *locality*. Thus, local interpolation in the source domain would ideally be similar to local interpolation in target domain when transferred.

There are often many possible alignments of semantic attributes that could maintain locality. For instance, absent additional context, there is no reason that dataset images and spoken utterances of the digit “0” should align with each other. There may also be no agreed common semantics, like for example between images of landscapes and passages of music, and it is at the liberty of the user to define such connections based on their own intent. Our goal in modeling is to respect such intent and make sure that the correct attributes are connected between the two domains.

We refer to this property as *semantic alignment*. A user can thus sort a set of data points from in each domain into common bins, which we can use to constrain the cross-domain alignment. We can quantitatively measure the degree of semantic alignment by using a classifier to label transformed data and measuring the percentage of data points that fall into the same bin for the source and target domain. Our goal can thus be stated as learning transformations that preserve locality and semantic alignment, while requiring as few labels from a user as possible.

To achieve this goal and tackle prior limitations, we propose to abstract the domains with independent latent variable models, and then learn to transfer between the latent spaces of those models. Our main contributions include:

- We propose a shared ”bridging” VAE to transfer between latent generative models. Locality and semantic alignment of transformations are encouraged by applying a sliced-wasserstein distance (SWD), and a classification loss respectively to the shared latent space.

- We demonstrate with qualitative and quantitative results that our proposed method enables transfer within a modality (image-to-image), between modalities (image-to-audio), and between generative model types (VAE to GAN).

- We show that decoupling the cost of training from that of the base generative models increases training speed by a factor of $\sim 200\times$, even for the relatively small base models examined in this work.

6.2 Related Work

Latent Generative Models: Deep latent generative models use an expressive neural network function to convert a tractable latent distribution $p(z)$ into the approximation of a population distribution $p^*(x)$. Two popular variants include VAEs [79] and GANs [55]. GANs are trained with an adversarial classifier while VAEs are trained to maximize a variational approximation through the use of evidence lower bound (ELBO). These classes of models have been thoroughly investigated in many applications and variants [59, 94, 17] including conditional generation [110], generation of one domain conditioned on another [36, 124], generation of high-quality images [77], audio [47, 49], and music [128].

Domain Transfer: Deep generative models enable domain transfer by learning a smooth mapping between data domains such that the variations in one domain are reflected in the other. This has been demonstrated to great effect within a single modality, for example transferring between two different styles of image [69, 173, 96, 95], video [156], or music [113]. These works have been the basis of interesting creative tools [4], as small changes in the source domain are reflected by comparable intuitive changes in the target domain.

Despite these successes, this line of work has several limitations. Supervised techniques such as Pix2Pix [69] and Vid2Vid [156], are able to transfer between more distant datasets, but require very dense supervision from large volumes of tightly paired data. While latent translation can benefit from additional supervision, it does not require all data to be strongly paired data. Unsupervised methods such as CycleGAN or its variants [173, 140] require the two data domains to be closely related (e.g. horse-to-zebra, face-to-emoji, MNIST-to-SVHN) [96]. This allows the model to focus on transferring local properties like texture and coloring instead of high-level semantics. [33] show that CycleGAN transformations share many similarities with adversarial examples, hiding information about the source domain in near-imperceptible high-frequency variations of the target domain. Latent translation avoids these issues by abstracting the data domains with pretrained models, allowing them to be significantly different.

Perhaps closest to this work is the UNIT framework [99, 100], where a shared latent space is learned jointly with both VAE and GAN losses. In a similar spirit, they tie the weights of highest layers of the encoders and decoders to encourage learning a common latent space. While UNIT is sufficient for image-to-image translation (dog-to-dog, digit-to-digit), this work extends to more diverse data domains by allowing independence of the base generative models and only learning a shared latent space to tie the two together. Also, while joint training has performance benefits for a single domain transfer task, the modularity of latent translation allows specifying new model combinations without the potentially prohibitive cost of retraining the base models for each new combination.

Transfer Learning: Transfer learning and fine-tuning aim to reuse a model trained on a specific task for new tasks. For example, deep classification models trained on the ImageNet dataset [131] can transfer their learned features to tasks such as object detection [125] and semantic segmentation [102]. Natural language processing has also recently seen significant progress through transfer learning of very large pretrained models [41]. However, easily combining multiple models together in a modular way is still an unsolved problem. This work explores a step in that direction for deep latent generative models.

Neural Machine Translation: Unsupervised neural machine translation (NMT) techniques work by aligning embeddings in two different languages. Many approaches use discriminators to make translated embeddings indistinguishable [169, 88], similar to applying CycleGAN in latent space. This work takes that approach as a baseline and expands upon it by learning a shared embedding space for each domain. Backtranslation and anchor words [89, 10] are promising developments in NMT, and exploring their relevance to latent translation of generative models is an interesting avenue for future work.

6.3 Method

Figure 6.1 diagrams our hierarchical approach to latent translation. We start with a separate pretrained generative model, either VAE or GAN, for both the source and target domain. These models give latent embeddings (z_1, z_2) of the data from both domains (x_1, x_2). For VAEs, data embeddings are given by the encoder, $z \sim q(z|x)$, where q is an encoder network. Since GANs lack an encoder, we choose latent samples from the prior, $z \sim p(z)$, and then use rejection sampling to keep only samples whose associated data, $x = g(z)$, where g is a generator network, is classified with high confidence by an aux-

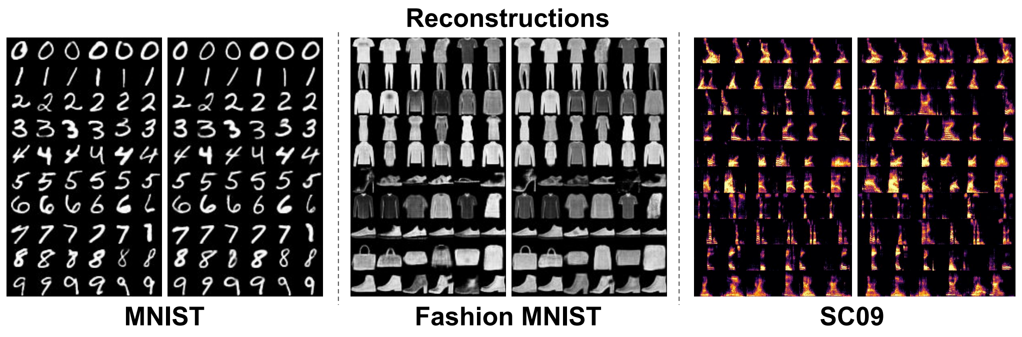


Figure 6.3: Bridging autoencoder reconstructions. For each dataset, original data is on the left and reconstructions are on the right. For SC09 we show the log magnitude spectrogram of the audio, and label order is the same as MNIST. The reconstruction quality is limited by the fidelity of the base generative model. The bridging autoencoder is able to achieve sharp reconstructions because the base models are either VAEs with $\beta < 1$ (MNIST, Fashion MNIST) or a GAN (SC09). More discussions are available in Section 6.5.1.

iliary attribute classifier. We then train a single "bridging" VAE to create a shared latent space (z') that corresponds to both domains. The bridging VAE shares weights between both latent domains to encourage the model to seek common structure, but we also find it helpful to condition both the encoder $q_{shared}(z'|z, D)$ and decoder $g_{shared}(z|z', D)$, with an additional one-hot domain label, D, to allow the model some flexibility to adapt to variations particular to each domain. While the base VAEs and GANs have spherical Gaussian priors, we penalize the KL-Divergence term for VAEs to be less than 1 (also known as a β-VAE [63]), allowing the models to achieve better reconstructions and retain some structure of the original dataset for the bridging VAE to model. Full architecture and training details are available in the Supplementary Material.

The domain conditional bridging VAE objective consists of three loss terms:

1. **Evidence Lower Bound (ELBO)**. Standard VAE loss. For each domain $d \in \{1, 2\}$,

$$\begin{aligned}\mathcal{L}_d^{\text{ELBO}} = & -\mathbb{E}_{z' \sim Z'_d}\left[\log \pi(z_d; g(z', D = d)\right] \\ & + \beta_{\text{KL}} D_{\text{KL}}\left(q(z'|z_d, D = d) \| p(z')\right)\end{aligned}$$

where the likelihood $\pi(z; g)$ is a spherical Gaussian $\mathcal{N}(z; g, \sigma^2 I)$, and σ and β_{KL} are hyperparmeters set to 1 and 0.1 respectively to encourage reconstruction accuracy.

2. **Sliced Wasserstein Distance (SWD)** [19]. The distribution distance between mini-batches of samples from each domain in the shared latent space (z'_1, z'_2),

$$\mathcal{L}^{\text{SWD}} = 1/|\Omega| \sum_{\omega \in \Omega} W_2^2 \left(\text{proj}(z'_1, \omega), \text{proj}(z'_2, \omega)\right)$$

where Ω is a set of random unit vectors, $\text{proj}(A, a)$ is the projection of A on vector a, and $W_2^2(A, B)$ is the quadratic Wasserstein distance.

3. **Classification Loss (Cls)**. For each domain $d \in \{1, 2\}$, we enforce semantic alignment with attribute labels y and a classification loss in the shared latent space:

$$\mathcal{L}_d^{\text{Cls}} = \mathbb{E}_{z' \in Z'_d} H(f(z'), y)$$

where H is the cross entropy loss, $f(z')$ is a linear classifier.

Including terms for both domains, it gives the total training loss,

$$\mathcal{L} = (\mathcal{L}_1^{\text{ELBO}} + \mathcal{L}_2^{\text{ELBO}}) + \beta_{\text{SWD}} \mathcal{L}^{\text{SWD}} + \beta_{\text{Cls}} (\mathcal{L}_1^{\text{Cls}} + \mathcal{L}_2^{\text{Cls}})$$

where β_{SWD} and β_{Cls} are scalar loss weights. The transfer procedure is illustrated Figure 6.2 using synthetic data. For reconstructions, data x_1 is passed through two encoders, $z_1 \sim q(z_1|x_1)$, $z' \sim q_{\text{shared}}(z'|z_1, D = 1)$, and two decoders, $\hat{z_1} \sim g_{\text{shared}}(\hat{z_1}|z', D = 1)$, $\hat{x_1} \sim g(\hat{x_1}|\hat{z_1})$. For transformations, the encoding is the same, but decoding uses decoders (and conditioning) from the second domain, $\hat{z_2} \sim g_{\text{shared}}(\hat{z_2}|z', D = 2)$, $\hat{x_2} \sim g(\hat{x_2}|\hat{z_2})$. Further analysis and intuition behind loss terms is given in Figure 6.2.

6.4 Experiments

6.4.1 Datasets

While the end goal of our method is to enable mapping between arbitrary datasets, for quantitative evaluation we restrict ourselves to three domains where there exist a somewhat natural alignment for comparison:

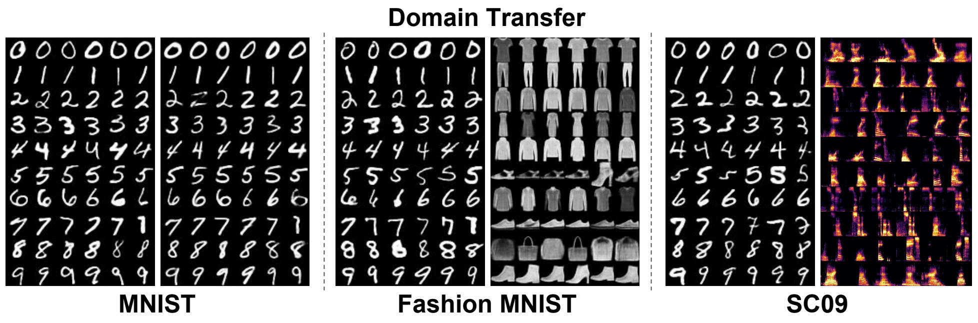


Figure 6.4: Domain Transfer from an MNIST VAE to a separately trained MNIST VAE, Fashion MNIST VAE, and SC09 GAN. For each dataset, the left is the data from the source domain and on the right is transformations to the target domain. Domain transfer maintains the label identity, and a diversity of outputs, mapping local variations in the source domain to local variations in the target domain. More discussions available in Section 6.5.2.

Transfer Accuracy						FID
Model Type	MNIST → MNIST	MNIST → F-MNIST	F-MNIST → MNIST	MNIST → SC09	SC09 → MNIST	MNIST → F-MNIST
Pix2Pix	-	0.77	0.08	×	×	0.087
CycleGAN	-	0.08	0.13	×	×	0.361
Latent CycleGAN	0.08	0.10	0.10	0.09	0.11	0.081
This work	**0.98**	**0.95**	**0.89**	**0.67**	**0.98**	**0.004**

Table 6.1: Quantitative comparison of domain transfer accuracy and quality. We compare to preexisting approaches, Pix2Pix [69] and CycleGAN [173] trained on raw-pixels, and latent tranlsation via CycleGAN (Latent CycleGAN). All baseline models are trained with pairs of class-aligned data. As in Figure 6.4, MNIST → MNIST transfers between pretrained models with different initial conditions. Pix2Pix and CycleGAN fail to train on MNIST → SC09, as the two domains are too distinct. We compute class accuracies and Fréchet Inception Distance (FID, lower value indicates a better image quality), using pretrained classifiers on the target domain. Latent CycleGAN accuracies are similar to chance because the cyclic reconstruction cost dominates and encourages learning the identity function. More discussions available in Section 6.5.2.

1. **MNIST** [90], which contains images of hand-written digits of 10 classes from "0" to "9".

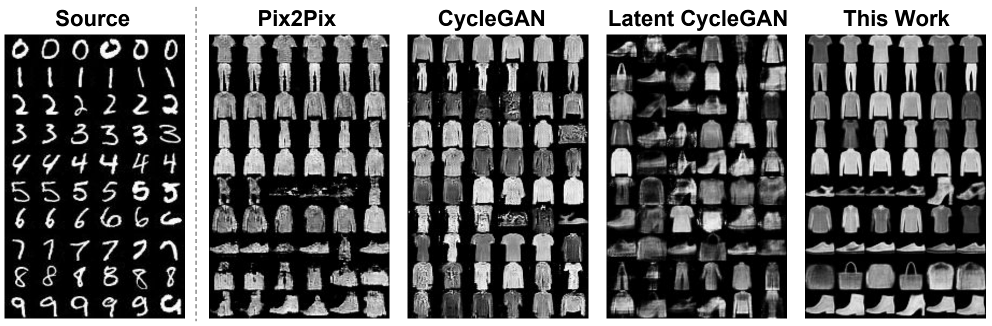


Figure 6.5: Qualitative comparison of domain transfer across models, as in the last column of Table 6.1. Pix2Pix seems to have collapsed to a prototype for each class and CycleGAN has collapsed to output mostly shirts and jackets. Latent CycleGAN also has no class structure, as compared to the bridging autoencoder that generates clear and diverse class-appropriate transformations. More discussions available in Section 6.5.2.

2. **Fashion MNIST** [163], which contains fashion related objects such as shoes, t-shirts, categorized into 10 classes. The structure of data and the size of images are identical to MNIST.

3. **SC09**, a subset of the Speech Commands Dataset [21], which contains spoken digits from "0" to "'9". This is a much noisier and more difficult dataset than the others, with 16,000 dimensions (1 second 16kHz) instead of 768. Since WaveGAN lacks an encoder, the bridging VAE dataset is composed of samples from the prior rather than encodings of the data. We collect 1,300 prior samples per a label by rejecting samples with a maximum softmax output < 0.95 on the pretrained WaveGAN classifier for spoken digits.

For MNIST and Fashion MNIST, we pretrain VAEs with MLP encoders and decoders following [48]. For SC09, we chose to use the publicly available WaveGAN [43][22] because we wanted a global latent variable for the full waveform (as opposed to a distributed latent code as in [49]) and VAEs fail on this more difficult dataset. It also gives us the opportunity to explore

[21]Dataset available at https://ai.googleblog.com/2017/08/launching-speech-commands-dataset.html

[22]Code and pretrained model available at https://github.com/chrisdonahue/wavegan

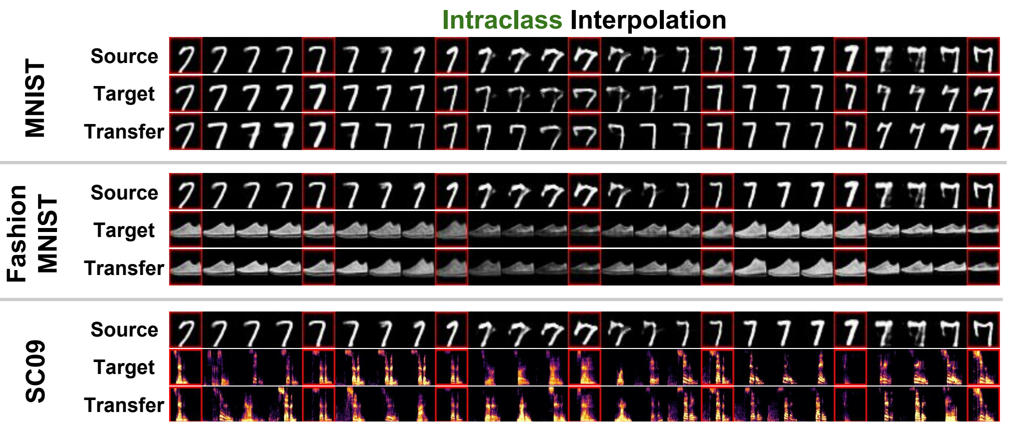


Figure 6.6: Interpolation within a class using bridging autoencoders. Columns of images in red squares are fixed points, with three rows of interpolations. (1) Source: Interpolate in source domain between fixed data points, (2) Target: Transfer fixed data points in source domain to target domain and interpolate between transferred fixed points there, (3) Transfer: Transfer all points in first row to the target domain. Note that Transfer interpolation produces smooth variation of data attributes like Target interpolation, indicating that local variations in the source domain are mapped to local variations in the target domain. More discussions are available in Section 6.5.3.

transferring between different classes of models. For the bridging VAE, we also use stacks of fully-connected linear layers with ReLU activation and a final "gated mixing layer". Full network architecture details are available in the Supplementary Material. We would like to emphasize that we only use *class level* supervision for enforcing semantic alignment with the latent classifier.

We examine three scenarios of domain transfer:

1. MNIST $\leftrightarrow$ MNIST. We first train two lower-level VAEs from different initial conditions. The bridging autoencoder is then tasked with transferring between latent spaces while maintaining the digit class from source to target.

2. MNIST $\leftrightarrow$ Fashion MNIST. In this scenario, we specify a global one-to-one mapping between 10 digit classes and 10 fashion object classes (available in the Supplementary Material) The bridging autoencoder is

tasked with preserving this mapping as it transfers between images of digits and clothing.

3. MNIST $\leftrightarrow$ SC09. For the speech dataset, we first train a GAN to generate audio waveforms [43] of spoken digits. The bridging autoencoder is then tasked with transferring between a VAE of written digits and a GAN of spoken digits.[23]

6.4.2 Baselines

Where possible, we compare latent translation with a bridging VAE to three existing approaches. As a straightforward baseline, we train Pix2Pix [69] and CycleGAN [173] models to perform domain transfer directly in the data space. In analogy to the NMT literature, we also provide a baseline of latent translation with a CycleGAN in latent space (Latent CycleGAN). To encourage semantic alignment in baselines, we provide additional supervision by only training on class aligned pairs of data.

6.5 Results

6.5.1 Reconstruction

For bridging VAEs, qualitative reconstruction results are shown in Figure 6.3. The quality is limited by the fidelity of the base generative model, which is quite sharp for MNIST and Fashion MNIST (VAEs with $\beta < 1$). However, despite being accurate and intelligible, the "ground truth" of WaveGAN samples for SC09 is quite noisy, as the model is not completely successful at capturing this relatively difficult dataset. For all datasets the bridging VAE produces reconstructions of comparable quality to the original data.

Quantitatively, we can calculate reconstruction accuracies as shown in Table 6.2. With pretrained classifiers in each data domain, we measure the percentage of reconstructions that do not change in their predicted class. As expected from the qualitative results, the bridging VAE reconstruction accuracies for MNIST and Fashion MNIST are similar to the base models, indicating that it has learned the latent data manifold.

[23]Generated audio samples available in the supplemental material and at https://drive.google.com/drive/folders/1mcke0IhLucWtlzRchdAleJxcaHS7R-Iu

The lower accuracy on SC09 likely reflects the worse performance of the base generative model on the difficult dataset, leading to greater variance in the latent space. For example, only 31.8% samples from the GAN prior give the classifier high enough confidence to be used for training the bridging VAE. Despite this, the model still achieves relatively high reconstruction accuracy.

Accuracy	MNIST	Fashion MNIST	SC09
Base Model	0.995	0.952	-
Bridging VAE	0.989	0.903	0.739

Table 6.2: Reconstruction accuracy for base generative models and bridging VAE. WaveGAN does not have an encoder, and thus does not have a base reconstruction accuracy.

6.5.2 Domain Transfer

For bridging VAEs, qualitative transfer results are shown in Figure 6.4. For each pair of datasets, domain transfer maintains the label identity, and a diversity of outputs, at a similar quality to the reconstructions in Figure 6.3.

Quantitative results are given in Table 6.1. Given that the datasets have pre-aligned classes, when evaluating transferring from data x_{d_1} to x_{d_2}, the reported accuracy is the portion of instances that x_{d_1} and x_{d_2} have the same predicted class. We also compute Fréchet Inception Distance (FID) as a measure of image quality using features form the same pretrained classifiers [62]. The bridging VAE achieves very high transfer accuracies (bottom row), which are comparable in each case to reconstruction accuracies in the target domain. Interestingly, it follows that MNIST $\rightarrow$ SC09 has lower accuracy than SC09 $\rightarrow$ MNIST, reflecting the reduced quality of the WaveGAN latent space.

In Table 6.1 we also quantitatively compare with the baseline models where possible. We exclude Pix2Pix and CycleGAN from MNIST $\rightarrow$ MNIST as it involves transfer between different initializations of the same model on the same dataset and does not apply. Despite a large hyperparameter search, Pix2Pix and CycleGAN fail to train on MNIST $\leftrightarrow$ SC09, as the two domains are too distinct. Qualitative results for MNIST $\rightarrow$ Fashion MNIST are shown in Figure 6.5. Pix2Pix has higher accuracy than the other baselines because

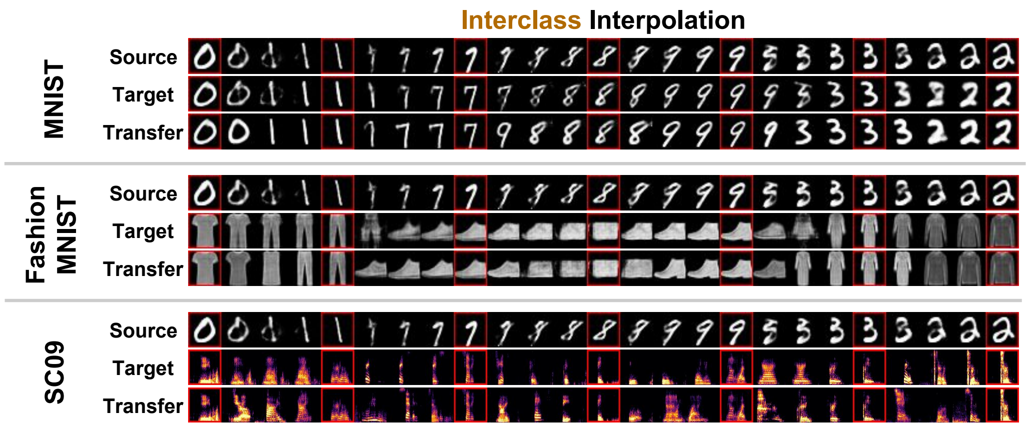


Figure 6.7: Interpolation between classes. The arrangement of images is the same as Figure 6.6. Source and Target interpolation produce varied, but sometimes unrealistic, outputs between class fixed points. The bridging autoencoder is trained to stay close to the marginal posterior of the data distribution. As a result Transfer interpolation varies smoothly within a class but makes larger jumps at class boundaries to avoid unrealistic outputs. More discussions are available in Section 6.5.3.

it seems to have collapsed to a prototype for each class, while CycleGAN has collapsed to outputs mostly shirts and jackets. In contrast, the bridging autoencoder generates diverse and class-appropriate transformations, with higher image quality which is reflected in their lower FID scores.

Finally, we found that latent CycleGAN had transfer accuracies roughly equal to chance. Unlike NMT, the base latent spaces have spherical Gaussian priors that make the cyclic reconstruction loss easy to optimize, encouraging generators to learn an identity function. Indeed, the samples in Figure 6.5 resemble samples from the Gaussian prior, which are randomly distributed in class. This motivates the need for new techniques for latent translation such as the bridging VAE in this work.

6.5.3 Interpolation

Interpolation can act as a good proxy for locality, as good interpolation requires small changes in the source domain to cause small changes in the target domain. We show intraclass and interclass interpolation in Figure 6.6 and Figure 6.7 respectively. We use spherical interpolation [68], as we are interpolating in a Gaussian latent space. The figures compare three types of interpolation: (1) the interpolation in the source domain's latent space, which acts a baseline for smoothness of interpolation for a pretrained generative model, (2) transfer fixed points to the target domain's latent space and interpolate in that space, and (3) transfer all points of the source interpolation to the target domain's latent space, which shows how the transferring warps the latent space.

Note for intraclass interpolation, the second and third rows have comparably smooth trajectories, reflecting that locality has been preserved. For interclass interpolation in Figure 6.7 interpolation is smooth within a class, but between classes the second row blurs pixels to create blurry combinations of digits, while the full transformation in the third row makes sudden transitions between classes. This is expected from our training procedure as the bridging autoencoder is modeling the marginal posterior of each latent space, and thus always stays on the manifold of the actual data during interpolation.

6.5.4 Efficiency and Ablation Analysis

Since our method is a semi-supervised method, we want to know how effectively our method leverages the labeled data. In Table 6.3 we show for the MNIST $\rightarrow$ MNIST setting the performance measured by transfer accuracy with respect to the number of labeled data points. Labels are distributed evenly among classes. The accuracy of transformations grows monotonically with the number of labels and reaches over 50% with as few as 10 labels per a class. Without labels, we also observe accuracies greater than chance due to unsupervised alignment introduced by the SWD penalty in the shared latent space.

Labels / Class	0	1	10	100	1000	6000
Accuracy	0.1390	0.339	0.524	0.6810	0.898	0.980

Table 6.3: MNIST $\rightarrow$ MNIST transfer accuracy with labeled data.

Besides data efficiency, pretraining the base generative models has computational advantages. For large generative models that take weeks to train, it would be infeasible to retrain the entire model for each new cross-domain mapping. The bridging autoencoder avoids this cost by only retraining the latent transfer mappings. As an example from these experiments, training the bridging autoencoder for MNIST $\leftrightarrow$ SC09 takes about half an hour on a single GPU, while retraining the SC09 WaveGAN takes around four days.

Finally, we perform an ablation study to confirm the benefits of each architecture component to transfer accuracy. For consistency, we stick to the MNIST $\rightarrow$ MNIST setting with fully labeled data. In Table 6.4, we see that the largest contribution to performance is from the domain conditioning signal, allowing the model to adapt to the specific structure of each domain. Further, the increased overlap in the shared latent space induced by the SWD penalty is reflected in the greater transfer accuracies.

Method Ablation	Unconditional VAE	Domain Conditional VAE	Domain Conditional VAE + SWD
Accuracy	0.149	0.849	0.980

Table 6.4: Ablation study of MNIST $\rightarrow$ MNIST transfer accuracies.

6.6 Conclusion

We have demonstrated an approach to learn mappings between disparate domains by bridging the latent codes of each domain with a shared autoencoder. We find bridging VAEs are able to achieve high transfer accuracies, smoothly map interpolations between domains, and even connect different model types (VAEs and GANs). Here, we have restricted ourselves to datasets with intuitive class-level mappings for the purpose of quantitative comparisons, however there are many interesting creative possibilities to apply these techniques between domains without a clear semantic alignment. Finally, as a step towards modular design, we combine two pretrained models to solve a new task for significantly less computational resources than retraining the models from scratch.

We expect that several future research directions exist following up this work. One direction is applying this method to real-world interactive art creativity where users can benefit from operating in one domain and get the creativity express in another hard or tedious to tune domain. This direction could be important since the art design presents a significant barrier to newcomers, and lowering the barrier by allowing creation in a more accessible domain to be reflected in another hard domain allows more involving of people in the creative process. Another essential direction concerns combining this work with structured data while maintaining meaningful structures in downstream domains. This work still focuses on continuous domains such as images and audio, which are easier to model with relatively good quality. However, how to deal with the cross-domain transfer while handling the complexity of structured data remains an exciting and open question for further study.

Chapter Appendix

6.A Training Target Design

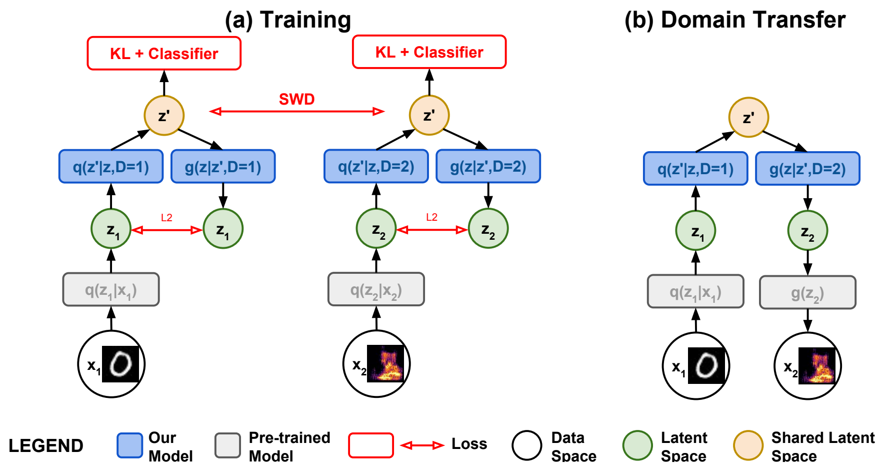


Figure 6.8: Architecture and training. Our method aims at transfer from one domain to another domain such that the correct semantics (e.g., label) is maintained across domains and local changes in the source domain should be reflected in the target domain. To achieve this, we train a model to transfer between the latent spaces of pre-trained generative models on source and target domains. **(a)** The training is done with three types of loss functions: (1) The VAE ELBO losses to encourage modeling of z_1 and z_2, which are denoted as L2 and KL in the figure. (2) The Sliced Wasserstein Distance loss to encourage cross-domain overlapping in the shared latent space, which is denoted as SWD. (3) The classification loss to encourage intraclass overlap in the shared latent space, which is denoted as Classifier. The training is semi-supervised, since (1) and (2) requires no supervision (classes) while only (3) needs such information. **(b)** To transfer data from one domain x_1 (an image of digit “0”) to another domain x_2 (an audio of human saying “zero”, shown in form of spectrum in the example), we first encode x_1 to $z_1 \sim q(z_1|x_1)$, which we then further encode to a shared latent vector z' using our conditional encoder, $z' \sim q(z'|z_1, D = 1)$, where D donates the operating domain. We then decode to the latent space of the target domain $z_2 = g(z|z', D = 2)$ using our conditional decoder, which finally is used to generate the transferred audio $x_2 = g(x_2|z_2)$.

We want to archive following three goals for the proposed VAE for latent spaces:

1. It should be able to model the latent space of both domains, including modeling local changes as well.
2. It should encode two latent spaces in a way to enable domain transferability. This means encoded z_1 and z_2 in the shared latent space should occupy overlapped spaces.
3. The transfer should be kept in the same class. That means, regardless of domain, zs for the same class should occupy overlapped spaces.

With these goals in mind, we propose to use an optimization target composing of three kinds of losses. In the following text for notational convenience, we denote approximated posterior $Z'_d \triangleq q(z'|z_d, D = d), z_d \sim q(z_d|x_d), x_d \sim p(x_d)$ for $d \in \{1, 2\}$, the process of sampling z'_d from domain d.

For reference, In Figure 6.9 we show the intuition to design and the contribution to performance from each loss terms. Also, the complete diagram of traing target is detailed in Figure 6.8.

1. Modeling two latent spaces with local changes. VAEs are often used to model data with local changes in mind, usually demonstrated with smooth interpolation, and we believe this property also applies when modeling the latent space of data. Consider for each domain $d \in \{1, 2\}$, the VAE is fit to data to maximize the ELBO (Evidence Lower Bound)

$$\begin{aligned}\mathcal{L}_d^{\text{ELBO}} = &- \mathbb{E}_{z' \sim Z'_d} \left[\log \pi(z_d; g(z', D = d)\right] \\ &+ \beta_{\text{KL}} D_{\text{KL}} \left(q(z'|z_d, D = d) \| p(z')\right)\end{aligned}$$

where both q and g are fit to maximize $\mathcal{L}_d^{\text{ELBO}}$. Notably, the latent space zs are continuous, so we choose the likelihood $\pi(z; g)$ to be the product of $\mathcal{N}(z; g, \sigma^2 I)$, where we set σ to be a constant that effectively sets $\log \pi(z; g) = ||z - g||_2$, which is the L2 loss in Figure 6.8 (a). Also, D_{KL} is denoted as KL loss in Figure 6.8 (a).

2. Cross-domain overlapping in shared latent space. Formally, we propose to measure the cross-domain overlapping through the distance between following two distributions as a proxy: the distribution of z' from source domain (e.g., $z'_1 \sim Z'_1$) and that from the target domain (e.g., $z'_2 \sim Z'_1$). We use Wasserstein Distance [7] to measure the distance of two sets of samples (this notion straightforwardly applies to the mini-batch setting) S'_1 and S'_2, where S'_1 is sampled from the source domain $z'_1 \sim Z'_1$ and S'_1 from the target domain $z'_2 \sim Z'_d$. For computational efficiency and inspired by [40], we use SWD, or Sliced Wasserstein Distance [19] between S'_1 and S'_2 as a loss term to encourage cross-domain overlapping in shared latent space. This means in practice we introduce the loss term

$$\mathcal{L}^{\text{SWD}} = \frac{1}{|\Omega|} \sum_{\omega \in \Omega} W_2^2 \left(\text{proj}(S'_1, \omega), \text{proj}(S'_2, \omega) \right)$$

where Ω is a set of random unit vectors, $\text{proj}(A, a)$ is the projection of A on vector a, and $W_2^2(A, B)$ is the quadratic Wasserstein distance, which in the one-dimensional case can be easily solved by monotonically pairing points in A and B, as proven in [40].

3. Intra-class overlapping in shared latent space. We want that regardless of domain, zs for the same class should occupy overlapped spaces, so that instance of a particular class should retain its label through the transferring. We therefore introduce the following loss term for both domain $d \in \{1, 2\}$

$$\mathcal{L}_d^{\text{Cls}} = \mathbb{E}_{z' \in Z'_d} H(f(z'), l_{x'})$$

where H is the cross entropy loss, $f(z')$ is a one-layer linear classifier, and $l_{x'}$ is the one-hot representation of label of x' where x' is the data associated with z'. We intentionally make classifier f as simple as possible in order to encourage more capacity in the VAE instead of the classifier. Notably, unlike previous two categories of losses that are unsupervised, this loss requires labels and is thus supervised.

6.B Model Architecture

6.B.1 Bridging VAE

The model architecture of our proposed Bridging VAE is illustrated in Figure 6.10. The model relies on Gated Mixing Layers, or GML. We find

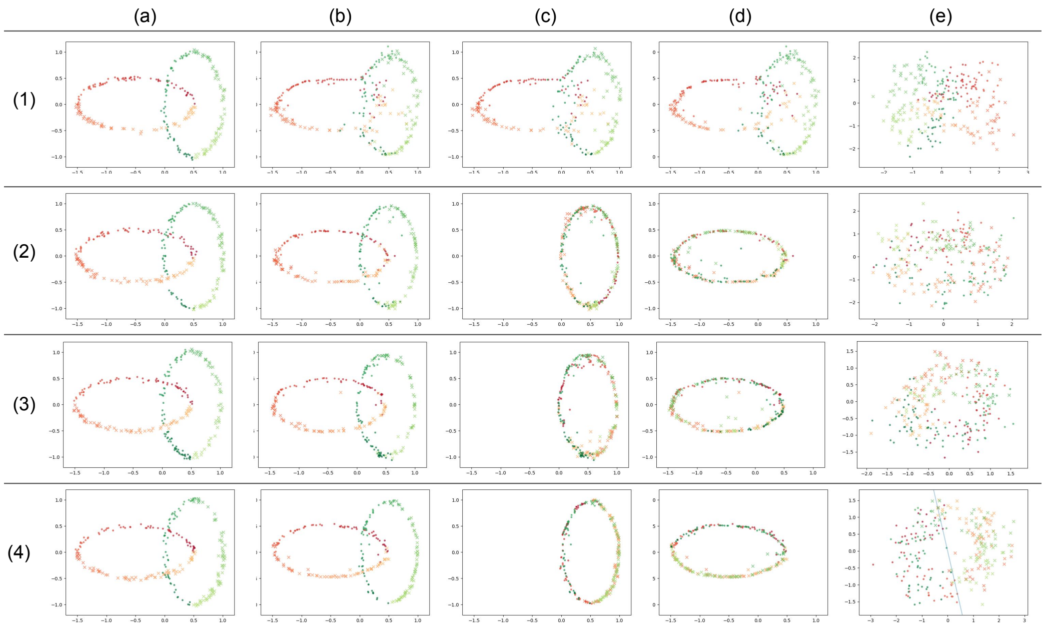


Figure 6.9: Synthetic data to demonstrate the transfer between 2-D latent spaces with 2-D shared latent space. Better viewed with color and magnifier. Columns (a) - (e) are synthetic data in latent space, reconstructed latent space points using VAE, domain 1 transferred to domain 2, domain 2 transferred to domain 1, shared latent space, respectively, follow the same arrangement as Figure 6.2. Each row represent a combination of our proposed components as follows: **(1)** Regular, unconditional VAE. Here transfer fails and the shared latent space are divided into region for two domains. **(2)** Conditional VAE. Here exists an overlapped shared latent space. However the shared latent space are not mixed well. **(3)** Conditional VAE + SWD. Here the shared latent space are well mixed, preserving the local changes across domain transfer. **(4)** Conditional + SWD + Classification. This is the best scenario that enables both domain transfer and class preservation as well as local changes. An overall observation is that each proposed component contributes positively to the performance in this synthetic data, which serves as a motivation for our decision to include all of them.

empirically that GML improves performance by a large margin than linear layers, for which we hypothesize that this is because both the latent space (z_1, z_2) and the shared latent space z' are Gaussian space, GML helps optimization by starting with a good initialization. We also explore other popular network components such as residual network and batch normalization, but find that they are not providing performance improvements. Also, the condition is fed to encoder and decoder as a 2-length one hot vector indicating one of two domains.

For all settings, we use the dimension of latent space 100, $\beta_{\text{SWD}} = 1.0$ and $\beta_{\text{CLs}} = 0.05$, Specifically, for MNIST $\leftrightarrow$ MNIST and MNIST $\leftrightarrow$ Fashion MNIST, we use the dimension of shared latent space 8, 4 layers of FC (fully Connected Layers) of size 512 with ReLU activation, $\beta_{\text{KL}} = 0.05$, $\beta_{\text{SWD}} = 1.0$ and $\beta_{\text{CLs}} = 0.05$; while for MNIST $\leftrightarrow$ SC09, we use the dimension of shared latent space 16, 8 layers of FC (fully Connected Layers) of size 1024 with ReLUa ctivation, $\beta_{\text{KL}} = 0.01$, $\beta_{\text{SWD}} = 3.0$ and $\beta_{\text{CLs}} = 0.3$. The difference is due to that GAN does not provide posterior, so the latent space points estimated by the classifier is much harder to model.

For optimization, we use Adam optimizer[78] with learning rate 0.001, $beta_1 = 0.9$ and $beta_2 = 0.999$. We train 50000 batches with batch size 128. We do not employ any other tricks for VAE training.

6.B.2 Base Models and Classifiers

Base Model for MNIST and Fashion-MNIST The base model for data space x (in our case MNIST and Fashion-MNIST) is a standard VAE with latent space z, consisting of an encoder function $q(z|x)$ that serves as an approximation to the posterior $p(z|x)$, a decoder function $g(z)$ and a likelihood $\pi\left(z; g(z)\right)$ that is defined as $p(z|x)$. Since we normalize MNIST and Fashion MNIST's pixel values such that they become continuous values in range $[0, 1]$, The likelihood $\pi\left(z; g\right)$ is accordingly Gaussian $\mathcal{N}\left(z; g, \sigma_x I\right)$. Both q and g are optimized to the Evidence Lower Bound (ELBO):

$$\mathcal{L}^{\text{ELBO}} = -\mathbb{E}_{x \sim X}\left[\log \pi(x; g(z)\right] + \beta D_{\text{KL}}\left(q(z|x) \| p(z)\right)$$

where $p(z)$ is a tractable simple prior which is $\mathcal{N}\left(0; I\right)$ We parameterize both encoder and decoder with neural networks. Specifically, the encoder consists of 3 layers of FC (Fully Connected Layers) of size 1024 with ReLU activation, on top of which are an affine transformation for z_μ and an FC with sigmoid activation for z_σ (both of size of latent space, which is 100), which give $q(z|x) \triangleq \mathcal{N}\left(z_\mu; z_\sigma\right)$. the decoder g consists of 3 layers of FC (Fully Connected Layers) of size 1024 with ReLU activation, on top of which is an affine transformation of size equaling number of pixels in data ($784 = 28 \times 28$, the size of MNIST and Fashion MNIST images).

For training details, we use $\beta = 1.0$ and $x_\sigma = 0.1$ for a better quality in reconstruction. we use Adam optimizer[78] with learning rate 0.001, $beta_1 = 0.9$ and $beta_2 = 0.999$ for optimization, and we train 100 epochs with batch size 512.

Base Classifier for MNIST and Fashion-MNIST The base classifier is consisting of 4 layers of FC followed by a affine transformation of size equaling to the number of labels (10). We use Adam optimizer[78] with learning rate 0.001, $beta_1 = 0.9$ and $beta_2 = 0.999$, and train for 100 epochs with batch size 256. We use the best classifier selected from dumps at end of each epoch, based on the performance on the hold-out set.

Base Model and Classifier for SC09 For SC09, we use the publicly available WaveGAN [43][24] that contains pretrained GAN model for generation and classifier for classification.

6.C Other Approach

An possible alternative approach exists that applies CycleGAN in the latent space. CycleGAN consists of two heterogeneous types of loss, the GAN and reconstruction loss, combined using a factor β that must be tuned. We found that adapting CycleGAN to the latent space rather than data space leads to quite different training dynamics and therefore thoroughly tuned β. Despite our effort, we notices that it leads to bad performance: transfer accuracy is 0.096 for MNIST $\rightarrow$ Fashion MNIST and 0.099 for Fashion MNIST $\rightarrow$ MNIST. We show qualitative results from applying CycleGAN in latent space in Figure 6.11.

6.D Supplementary Figures

We show the MNIST Digits to Fashion MNIST class Mapping we used in the MNIST $\leftrightarrow$ Fashion MNIST setting detailed in the experiments in Figure 6.5.

[24]https://github.com/chrisdonahue/wavegan

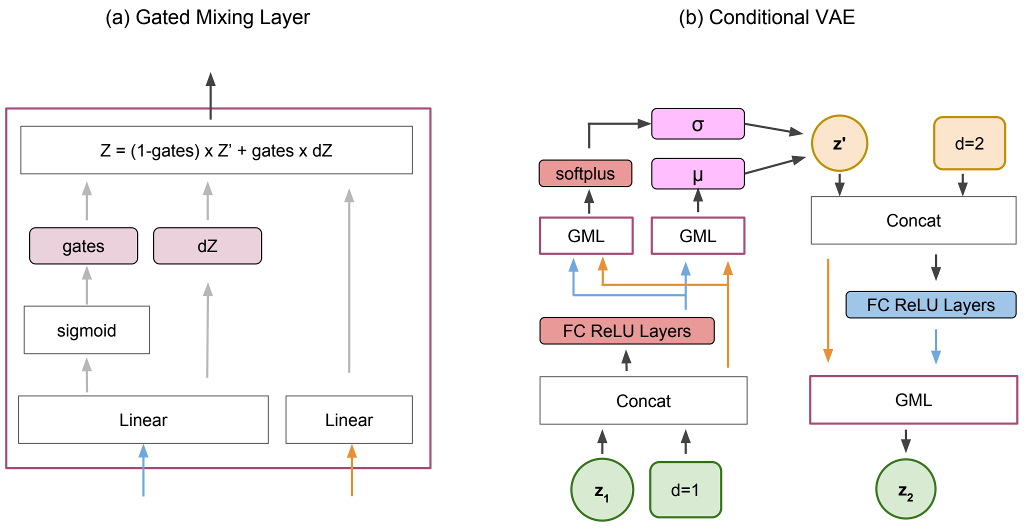


Figure 6.10: Model Architecture for our Conditional VAE. **(a)** Gated Mixing Layer, or GML, as an important building component. **(b)** Our conditional VAE with GML.

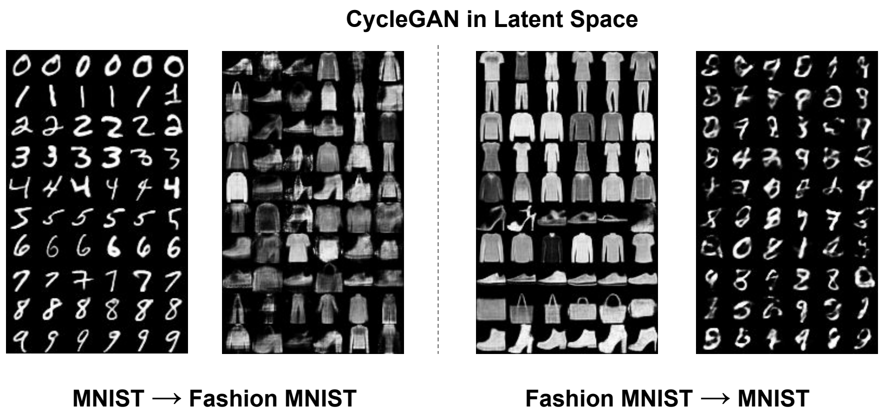


Figure 6.11: Qualitative results from applying CycleGAN [173] in latent space. Visually, this approach suffer from less desirable overall visual quality, and most notably, failure to respect label-level supervision, compared to our proposed approach.

MNIST Digits	Fashion MNIST Class
0	T-shirt/top
1	Trouser
2	Pullover
3	Dress
4	Coat
5	Sandal
6	Shirt
7	Sneaker
8	Bag
9	Ankle boot

Table 6.5: MNIST Digits to Fashion MNIST class Mapping, made according to Labels information available at https://github.com/zalandoresearch/fashion-mnist

7 Conclusion

The contributions of this thesis are organized following the conception of representation learning for different modalities of data under different circumstances in Chapter 1. In detail, the individual contributions, summarized as a list of different modularities and the proposed methods to apply representation learning on them, are:

- In Chapter 2, a generative model for structured data that captures the representations for discrete structures with formal grammars and semantics and generate both syntactically and semantically correct data.
- In Chapter 3, a fast and effective label inference for social relations in collaboration networks which can be formalized as a multi-label classification problem on graph edges.
- In Chapter 4, generating high-quality facial images of anime characters using generative adversarial network that are highly welcome by the community through the availability of a web interface that runs on edge devices.
- In Chapter 5, modeling bilingual dictionaries, or cross-lingual correspondence between sentences and lexicons, to benefit cross-lingual applications such as cross-lingual semantic search and question answering.
- In Chapter 6, a novel approach to bridge modularities by learning a post-hoc interface between two existing models to solve a new task and facilitate transferring across different modalities (e.g., image-to-audio) and even different types of generative models.

The approaches covered in this these deal with data of different modalities. Despite the differences in modalities, the approaches still could provide a unified underlying way to deal with the data through operating in the latent, continuous space.

To summarize, all there approaches share a common property of enjoying benefits from the fact that representation learning serves two purposes: one the one hand, latent representation, as a posterior, serves as features proven to be useful for downstream tasks as inputs, moreover, one the other hand, the latent space itself, as a prior, maintains a meaningful local structure which enables applications that require controlling the generation of data through manipulating the latent representation. For example, predicting the properties of data instances using latent representations as features works for graphs (Chapter 2), social networks (Chapter 3) sentences (Chapter 5), while control the generation of data through manipulating the latent representation works for images (Chapter 4) or even multiple modalities that are handled simultaneously (Chapter 6).

It is worth looking into how the two purposes, predictivity as features and manipulation through latent structures, are served through representation. First, usefully features are not bound to have suitable structures, which has been demonstrated by the performance of features learning algorithms without localities such as Random Projection [16] and Hashing [158], and adapting algorithm with a well-formed theoretical structure such as Manifold Learning [155, 97] to archive state-of-the-art empirical result remains an open research direction. Still, in the case of representation learning, both purposed are served simultaneously without fighting with each other, which suggests that at least the representation learning can optimize the serving of two purposes in a non-exclusive fashion. Furthermore, we observe that some applications, such as unsupervised machine translation [87, 10] and post-hocing generative models [48], benefit from having both good predictivities as features and latent structures. We, therefore, envision that these two purposes could, as one step forward, helping each other under the framing of representation learning.

7.1 Future Works

The contributions I present in this thesis show a line of approaches tackling the challenge on how to model data from different modality and how to model them in a unified way for downstream tasks to leverage. Besides detailed discussion and future work section in each Chapter, I also describe here, in a nutshell, some concrete future works for which this line of approaches opens a new door of research:

- The generative model for structured data in Chapter 2 tackles the challenge of addressing both syntax and semantic constraints in the generative model for structured data. This model is shown to empirically provide consistent and significant improvements over previous models without adding observable extra computation cost. The future work includes exploring the refinement of formalization on a more theoretical ground, and investigating the application of such formalization on a more diverse set of data modality, including computer programs and other molecule datasets.

- The inferring social relations in collaboration networks in Chapter 3 is formulated as a semi-supervised learning problem on graphs where edges have multiple labels. This simple approach is consistency better and thousands of times faster than previous methods on one series of social relation dataset. The future work calls for exploring principled indicator that characterizes a class of datasets that can benefit from this simple approach, as well as theoretical analysis of the advantages brought by our approaches.

- The anime face generation with high quality in Chapter 4 opens the door to community-driven creation powered by a deep generative method. This method shows a certain degree of customization can be feasibly materialized for one user to create artwork according to his/her liking. Future works include the finer control-ability of and interactivity between the users.

- The neural embedding model that captures the correspondence of cross-lingual lexical and sentential semantics in Chapter 5 are effective at enhancing the cross-lingual learning with limited resources and achieves promising performance on cross-lingual tasks. An important direction of future work is to explore whether the lexicon-sentence alignment can improve bilingual word embeddings. Applying BilDRL to bilingual question answering and semantic search systems is another important direction.

- The approach to learning mappings between disparate domains by bridging the latent codes of each domain with a shared autoencoder in Chapter 6 can achieve high transfer accuracies, smoothly map interpolations between domains, and even connect different model types (VAEs and GANs). Future works include applying this method to real-world interactive art creativity where users can benefit from operating in one domain and get the creativity express in another hard or tedious to tune domain.

References

[1] Andrew Aitken, Christian Ledig, Lucas Theis, Jose Caballero, Zehan Wang, and Wenzhe Shi. Checkerboard artifact free sub-pixel convolution: A note on sub-pixel convolution, resize convolution and convolution resize. *arXiv preprint arXiv:1707.02937*, 2017.

[2] Rami Al-Rfou, Bryan Perozzi, and Steven Skiena. Plyglot: Distributed word representations for multilingual nlp. In *The SIGNLL Conference on Computational Natural Language Learning*, 2013.

[3] David Alvarez-Melis and Tommi S Jaakkola. Tree-structured decoding with doubly-recurrent neural networks. *International Conference on Learning Representations*, 2017. accepted as poster.

[4] Zaid Alyafeai. Online pix2pix. https://zaidalyafeai.github.io/pix2pix/cats.html, 2018. Accessed: 2019-01-20.

[5] Mohammad Ruhul Amin, Alisa Yurovsky, Yingtao Tian, and Steven Skiena. Deepannotator: Genome annotation with deep learning. In *Proceedings of the 2018 ACM International Conference on Bioinformatics, Computational Biology, and Health Informatics*, BCB '18, pages 254–259, New York, NY, USA, 2018. ACM.

[6] Martin Arjovsky and Léon Bottou. Towards principled methods for training generative adversarial networks. *arXiv preprint arXiv:1701.04862*, 2017.

[7] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein gan. *arXiv preprint arXiv:1701.07875*, 2017.

[8] Sanjeev Arora, Rong Ge, Yingyu Liang, Tengyu Ma, and Yi Zhang. Generalization and equilibrium in generative adversarial nets (gans). *arXiv preprint arXiv:1703.00573*, 2017.

[9] Mikel Artetxe, Gorka Labaka, and Eneko Agirre. Learning principled bilingual mappings of word embeddings while preserving monolingual invariance. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2289–2294, 2016.

[10] Mikel Artetxe, Gorka Labaka, Eneko Agirre, and Kyunghyun Cho. Unsupervised neural machine translation. In *International Conference on Learning Representations*, 2018.

[11] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. In *International Conference on Learning Representations*, 2015.

[12] Colin Bannard and Chris Callison-Burch. Paraphrasing with bilingual parallel corpora. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 597–604. Association for Computational Linguistics, 2005.

[13] Marc G Bellemare, Ivo Danihelka, Will Dabney, Shakir Mohamed, Balaji Lakshminarayanan, Stephan Hoyer, and Rémi Munos. The cramer distance as a solution to biased wasserstein gradients. *arXiv preprint arXiv:1705.10743*, 2017.

[14] Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1798–1828, 2013.

[15] Mostapha Benhenda. Chemgan challenge for drug discovery: can AI reproduce natural chemical diversity? *CoRR*, abs/1708.08227, 2017.

[16] Ella Bingham and Heikki Mannila. Random projection in dimensionality reduction: applications to image and text data. In *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 245–250. ACM, 2001.

[17] Mikołaj Biǹkowski, Dougal J. Sutherland, Michael Arbel, and Arthur Gretton. Demystifying MMD GANs. In *International Conference on Learning Representations*, 2018.

[18] Esben Jannik Bjerrum. SMILES enumeration as data augmentation for neural network modeling of molecules. *CoRR*, abs/1703.07076, 2017.

[19] Nicolas Bonneel, Julien Rabin, Gabriel Peyré, and Hanspeter Pfister. Sliced and radon wasserstein barycenters of measures. *Journal of Mathematical Imaging and Vision*, 51(1):22–45, 2015.

[20] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. Translating embeddings for modeling multi-relational data. In *Advances in neural information processing systems*, pages 2787–2795, 2013.

[21] Samuel R. Bowman, Luke Vilnis, Oriol Vinyals, Andrew M. Dai, Rafal Józefowicz, and Samy Bengio. Generating sentences from a continuous space. In *Proceedings of the 20th SIGNLL Conference on Computational Natural Language Learning, CoNLL 2016, Berlin, Germany, August 11-12, 2016*, pages 10–21, 2016.

[22] Joan Bresnan, Ronald M Kaplan, Stanley Peters, and Annie Zaenen. Cross-serial dependencies in dutch. *Linguistic Inquiry*, 1982.

[23] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale GAN training for high fidelity natural image synthesis. *CoRR*, abs/1809.11096, 2018.

[24] Shaosheng Cao, Wei Lu, and Qiongkai Xu. Grarep: Learning graph representations with global structural information. In *Proceedings of the 24th ACM International Conference on Information and Knowledge Management, CIKM 2015, Melbourne, VIC, Australia, October 19 - 23, 2015*, pages 891–900, 2015.

[25] Sarath Chandar, Stanislas Lauly, Hugo Larochelle, Mitesh Khapra, Balaraman Ravindran, Vikas C Raykar, and Amrita Saha. An autoencoder approach to learning bilingual word representations. In *Advances in Neural Information Processing Systems*, pages 1853–1861, 2014.

[26] Haochen Chen, Xiaofei Sun, Yingtao Tian, Bryan Perozzi, Muhao Chen, and Steven Skiena. Enhanced network embeddings via exploiting edge labels. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management, CIKM 2018, Torino, Italy, October 22-26, 2018*, pages 1579–1582, 2018.

[27] Muhao Chen, Chang Ping Meng, Gang Huang, and Carlo Zaniolo. Neural article pair modeling for wikipedia sub-article machine. In *ECML-PKDD*, 2018.

[28] Muhao Chen, Yingtao Tian, Kai-Wei Chang, Steven Skiena, and Carlo Zaniolo. Co-training embeddings of knowledge graphs and entity descriptions for cross-lingual entity alignment. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018, July 13-19, 2018, Stockholm, Sweden.*, pages 3998–4004, 2018.

[29] Muhao Chen, Yingtao Tian, Xuelu Chen, Zijun Xue, and Carlo Zaniolo. On2vec: Embedding-based relation prediction for ontology population. In *Proceedings of the 2018 SIAM International Conference on Data Mining, SDM 2018, May 3-5, 2018, San Diego Marriott Mission Valley, San Diego, CA, USA.*, pages 315–323, 2018.

[30] Muhao Chen, Yingtao Tian, Mohan Yang, and Carlo Zaniolo. Multilingual knowledge graph embeddings for cross-lingual knowledge alignment. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI 2017, Melbourne, Australia, August 19-25, 2017*, pages 1511–1517, 2017.

[31] Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder–decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734, 2014.

[32] Sumit Chopra, Michael Auli, and Alexander M Rush. Abstractive sentence summarization with attentive recurrent neural networks. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 93–98, 2016.

[33] Casey Chu, Andrey Zhmoginov, and Mark Sandler. Cyclegan, a master of steganography. *CoRR*, abs/1712.02950, 2017.

[34] Alexis Conneau, Douwe Kiela, Holger Schwenk, Loïc Barrault, and Antoine Bordes. Supervised learning of universal sentence representations from natural language inference data. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 670–680, 2017.

[35] Jocelyn Coulmance, Jean-Marc Marty, Guillaume Wenzek, and Amine Benhalloum. Trans-gram, fast cross-lingual word-embeddings. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1109–1113, 2015.

[36] Bo Dai, Sanja Fidler, Raquel Urtasun, and Dahua Lin. Towards diverse and natural image descriptions via a conditional gan. In *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

[37] Hanjun Dai, Bo Dai, and Le Song. Discriminative embeddings of latent variable models for structured data. In *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016*, pages 2702–2711, 2016.

[38] Hanjun Dai, Bo Dai, Yan-Ming Zhang, Shuang Li, and Le Song. Recurrent hidden semi-markov model. *International Conference on Learning Representations*, 2017.

[39] Dipanjan Das and Noah A Smith. Paraphrase identification as probabilistic quasi-synchronous recognition. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing*, pages 468–476. Association for Computational Linguistics, 2009.

[40] Ishan Deshpande, Ziyu Zhang, and Alexander Schwing. Generative modeling using the sliced wasserstein distance. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3483–3491, 2018.

[41] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805, 2018.

[42] Jacob Devlin, Rabih Zbib, Zhongqiang Huang, Thomas Lamar, Richard Schwartz, and John Makhoul. Fast and robust neural network joint models for statistical machine translation. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 1370–1380, 2014.

[43] Chris Donahue, Julian McAuley, and Miller Puckette. Adversarial audio synthesis. In *International Conference on Learning Representations*, 2019.

[44] Li Dong and Mirella Lapata. Language to logical form with neural attention. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 1: Long Papers*, 2016.

[45] David K Duvenaud, Dougal Maclaurin, Jorge Iparraguirre, Rafael Bombarell, Timothy Hirzel, Alán Aspuru-Guzik, and Ryan P Adams. Convolutional networks on graphs for learning molecular fingerprints. In *Advances in Neural Information Processing Systems*, pages 2215–2223, 2015.

[46] Ben Eisner, Tim Rocktäschel, Isabelle Augenstein, Matko Bošnjak, and Sebastian Riedel. emoji2vec: Learning emoji representations from their description. *arXiv preprint arXiv:1609.08359*, 2016.

[47] Jesse Engel, Kumar Krishna Agrawal, Shuo Chen, Ishaan Gulrajani, Chris Donahue, and Adam Roberts. GANSynth: Adversarial neural audio synthesis. In *International Conference on Learning Representations*, 2019.

[48] Jesse Engel, Matthew Hoffman, and Adam Roberts. Latent constraints: Learning to generate conditionally from unconditional generative models. In *International Conference on Learning Representations*, 2018.

[49] Jesse Engel, Cinjon Resnick, Adam Roberts, Sander Dieleman, Mohammad Norouzi, Douglas Eck, and Karen Simonyan. Neural audio synthesis of musical notes with WaveNet autoencoders. In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 1068–1077, International Convention Centre, Sydney, Australia, 06–11 Aug 2017. PMLR.

[50] Peter Ertl and Ansgar Schuffenhauer. Estimation of synthetic accessibility score of drug-like molecules based on molecular complexity and fragment contributions. *Journal of cheminformatics*, 1(1):8, 2009.

[51] Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. LIBLINEAR: A library for large linear classification. *Journal of Machine Learning Research*, 9:1871–1874, 2008.

[52] Manaal Faruqui and Chris Dyer. Improving vector space word representations using multilingual correlation. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics*, pages 462–471, 2014.

[53] Xiaocheng Feng, Xiachong Feng, Bing Qin, Zhangyin Feng, and Ting Liu. Improving low resource named entity recognition using cross-lingual knowledge transfer. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, 2018.

[54] Rafael Gómez-Bombarelli, David Duvenaud, José Miguel Hernández-Lobato, Jorge Aguilera-Iparraguirre, Timothy D Hirzel, Ryan P Adams, and Alán Aspuru-Guzik. c. *arXiv preprint arXiv:1610.02415*, 2016.

[55] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.

[56] Stephan Gouws, Yoshua Bengio, and Greg Corrado. Bilbowa: Fast bilingual distributed representations without word alignments. In *Proceedings of the 32nd International Conference on Machine Learning*, pages 748–756, 2015.

[57] Aditya Grover and Jure Leskovec. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13-17, 2016*, pages 855–864, 2016.

[58] Gabriel Lima Guimaraes, Benjamin Sanchez-Lengeling, Pedro Luis Cunha Farias, and Alán Aspuru-Guzik. Objective-reinforced generative adversarial networks (ORGAN) for sequence generation models. *CoRR*, abs/1705.10843, 2017.

[59] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron C Courville. Improved training of wasserstein gans. In

I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5767–5777. Curran Associates, Inc., 2017.

[60] Koichi Hamada, Kentaro Tachibana, Tianqi Li, Hiroto Honda, and Yusuke Uchida. Full-body high-resolution anime generation with progressive structure-conditional generative adversarial networks. In *The European Conference on Computer Vision (ECCV) Workshops*, September 2018.

[61] Hanjun Dai* and Yingtao Tian* , Bo Dai, Steven Skiena, and Le Song. Syntax-directed variational autoencoder for structured data. In *International Conference on Learning Representations*, 2018.

[62] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *Advances in Neural Information Processing Systems*, pages 6626–6637, 2017.

[63] Irina Higgins, Loic Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. beta-vae: Learning basic visual concepts with a constrained variational framework. In *International Conference on Learning Representations*, 2016.

[64] Felix Hill, KyungHyun Cho, Anna Korhonen, and Yoshua Bengio. Learning to understand phrases by embedding the dictionary. *Transactions of the Association for Computational Linguistics*, 4:17–30, 2016.

[65] Baotian Hu, Zhengdong Lu, Hang Li, and Qingcai Chen. Convolutional neural network architectures for matching natural language sentences. In *Advances in neural information processing systems*, pages 2042–2050, 2014.

[66] Zhiting Hu, Xuezhe Ma, Zhengzhong Liu, Eduard H. Hovy, and Eric P. Xing. Harnessing deep neural networks with logic rules. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 1: Long Papers*, 2016.

[67] Kejun Huang, Matt Gardner, Evangelos Papalexakis, Christos Faloutsos, Nikos Sidiropoulos, Tom Mitchell, Partha P Talukdar, and Xiao Fu. Translation invariant word embeddings. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1084–1088, 2015.

[68] Ferenc Huszr. Gaussian distributions are soap bubbles. https://www.inference.vc/high-dimensional-gaussian-distributions-are-soap-bubble/, 2017. Accessed: 2019-01-20.

[69] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A. Efros. Image-to-image translation with conditional adversarial networks. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[70] Dave Janz, Jos van der Westhuizen, Brooks Paige, Matt Kusner, and Jose Miguel Hernandez Lobato. Learning a generative model for validity in complex discrete structures. *International Conference on Learning Representations*, 2018. accepted as poster.

[71] David Janz, Jos van der Westhuizen, and José Miguel Hernández-Lobato. Actively learning what makes a discrete sequence valid. *CoRR*, abs/1708.04465, 2017.

[72] Guoliang Ji, Kang Liu, Shizhu He, Jun Zhao, et al. Distant supervision for relation extraction with sentence-level attention and entity descriptions. In *Proceedings of the AAAI International Conference on Artificial Intelligence*, pages 3060–3066, 2017.

[73] Jyun-Yu Jiang, Francine Chen, Yan-Ying Chen, and Wei Wang. Learning to disentangle interleaved conversational threads with a siamese hierarchical network and similarity ranking. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1812–1822. Association for Computational Linguistics, 2018.

[74] Wengong Jin, Regina Barzilay, and Tommi Jaakkola. Junction tree variational autoencoder for molecular graph generation. *arXiv preprint arXiv:1802.04364*, 2018.

[75] Yanghua Jin, Jiakai Zhang, Minjun Li, Yingtao Tian, Huachun Zhu, and Zhihao Fang. Towards the automatic anime characters creation with generative adversarial networks. *CoRR*, abs/1708.05509, 2017.

[76] Nal Kalchbrenner, Edward Grefenstette, Phil Blunsom, Dimitri Kartsaklis, Nal Kalchbrenner, Mehrnoosh Sadrzadeh, Nal Kalchbrenner, Phil Blunsom, Nal Kalchbrenner, and Phil Blunsom. A convolutional neural network for modelling sentences. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, pages 212–217. Association for Computational Linguistics, 2014.

[77] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of GANs for improved quality, stability, and variation. In *International Conference on Learning Representations*, 2018.

[78] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[79] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. In *International Conference on Learning Representations*, 2014.

[80] Ryan Kiros, Yukun Zhu, Ruslan R Salakhutdinov, Richard Zemel, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. Skip-thought vectors. In *Advances in neural information processing systems*, pages 3294–3302, 2015.

[81] Donald E Knuth. Semantics of context-free languages. *Theory of Computing Systems*, 2(2):127–145, 1968.

[82] Naveen Kodali, Jacob Abernethy, James Hays, and Zsolt Kira. How to train your dragan. *arXiv preprint arXiv:1705.07215*, 2017.

[83] Philipp Koehn. Europarl: A parallel corpus for statistical machine translation. In *MT summit*, volume 5, pages 79–86, 2005.

[84] Vivek Kulkarni, Yingtao Tian, Parth Dandiwala, and Steven Skiena. Simple neologism based domain independent models to predict year of authorship. In *Proceedings of the 27th International Conference on Computational Linguistics, COLING 2018, Santa Fe, New Mexico, USA, August 20-26, 2018*, pages 202–212, 2018.

[85] Matt J. Kusner, Brooks Paige, and José Miguel Hernández-Lobato. Grammar variational autoencoder. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, pages 1945–1954, 2017.

[86] Guillaume Lample, Alexis Conneau, Ludovic Denoyer, Hervé Jégou, et al. Word translation without parallel data. In *International Conference on Learning Representations*, 2018.

[87] Guillaume Lample, Alexis Conneau, Ludovic Denoyer, and Marc'Aurelio Ranzato. Unsupervised machine translation using monolingual corpora only. *arXiv preprint arXiv:1711.00043*, 2017.

[88] Guillaume Lample, Alexis Conneau, Marc'Aurelio Ranzato, Ludovic Denoyer, and Herv Jgou. Word translation without parallel data. In *International Conference on Learning Representations*, 2018.

[89] Guillaume Lample, Myle Ott, Alexis Conneau, Ludovic Denoyer, and Marc'Aurelio Ranzato. Phrase-based & neural unsupervised machine translation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 5039–5049. Association for Computational Linguistics, 2018.

[90] Yann LeCun and Corinna Cortes. MNIST handwritten digit database. *online*, 2010.

[91] Christian Ledig, Lucas Theis, Ferenc Huszár, Jose Caballero, Andrew Cunningham, Alejandro Acosta, Andrew Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, et al. Photo-realistic single image super-resolution using a generative adversarial network. *arXiv preprint arXiv:1609.04802*, 2016.

[92] Jie Lei. Animegan. https://github.com/jayleicn/animeGAN, 2017.

[93] Tao Lei, Wengong Jin, Regina Barzilay, and Tommi S. Jaakkola. Deriving neural architectures from sequence and graph kernels. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, pages 2024–2033, 2017.

[94] Chun-Liang Li, Wei-Cheng Chang, Yu Cheng, Yiming Yang, and Barnabás Póczos. Mmd gan: Towards deeper understanding of moment matching network. In *Advances in Neural Information Processing Systems*, pages 2203–2213, 2017.

[95] Jerry Li. Twin-gan–unpaired cross-domain image translation with weight-sharing gans. *arXiv preprint arXiv:1809.00946*, 2018.

[96] Minjun Li, Haozhi Huang, Lin Ma, Wei Liu, Tong Zhang, and Yugang Jiang. Unsupervised image-to-image translation with stacked cycle-consistent adversarial networks. In *The European Conference on Computer Vision (ECCV)*, September 2018.

[97] Tong Lin and Hongbin Zha. Riemannian manifold learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(5):796–809, 2008.

[98] Jingzhou Liu, Wei-Cheng Chang, Yuexin Wu, and Yiming Yang. Deep learning for extreme multi-label text classification. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 115–124. ACM, 2017.

[99] Ming-Yu Liu, Thomas Breuel, and Jan Kautz. Unsupervised image-to-image translation networks. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 700–708. Curran Associates, Inc., 2017.

[100] Ming-Yu Liu and Oncel Tuzel. Coupled generative adversarial networks. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems 29*, pages 469–477. Curran Associates, Inc., 2016.

[101] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *The IEEE International Conference on Computer Vision (ICCV)*, December 2015.

[102] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3431–3440, 2015.

[103] Ang Lu, Weiran Wang, Mohit Bansal, Kevin Gimpel, and Karen Livescu. Deep multilingual correlation for improved word embeddings. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 250–256, 2015.

[104] Thang Luong, Hieu Pham, and Christopher D Manning. Bilingual word representations with monolingual quality in mind. In *Proceedings of the 1st Workshop on Vector Space Modeling for Natural Language Processing*, pages 151–159, 2015.

[105] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(Nov):2579–2605, 2008.

[106] Mattya. chainer-dcgan. https://github.com/mattya/chainer-DCGAN, 2015.

[107] Mattya. chainer-gan-lib. https://github.com/pfnet-research/chainer-gan-lib, 2017.

[108] Tomas Mikolov, Quoc V Le, and Ilya Sutskever. Exploiting similarities among languages for machine translation. *arXiv preprint arXiv:1309.4168*, 2013.

[109] Tomas Mikolov, Ilya Sutskever, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States.*, pages 3111–3119, 2013.

[110] Mehdi Mirza and Simon Osindero. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014.

[111] Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In *International Conference on Machine Learning*, pages 1928–1937, 2016.

[112] Aditya Mogadala and Achim Rettinger. Bilingual word embeddings from parallel and non-parallel corpora for cross-language text classification. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 692–702, 2016.

[113] Noam Mor, Lior Wolf, Adam Polyak, and Yaniv Taigman. A universal music translation network. *arXiv preprint arXiv:1805.07848*, 2018.

[114] Jonas Mueller and Aditya Thyagarajan. Siamese recurrent architectures for learning sentence similarity. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 2786–2792, 2016.

[115] Muhao Chen* and Yingtao Tian* , Haochen Chen, Kai-Wei Chang, Steven Skiena, and Carlo Zaniolo. Learning to represent bilingual dictionaries. In *Submitted*, 2019.

[116] nagadomi. waifu2x. https://github.com/nagadomi/waifu2x, 2015.

[117] Ani Nenkova and Kathleen McKeown. A survey of text summarization techniques. In *Mining text data*, pages 43–76. Springer, 2012.

[118] Augustus Odena, Christopher Olah, and Jonathon Shlens. Conditional image synthesis with auxiliary classifier gans. *arXiv preprint arXiv:1610.09585*, 2016.

[119] Emilio Parisotto, Abdel-rahman Mohamed, Rishabh Singh, Lihong Li, Dengyong Zhou, and Pushmeet Kohli. Neuro-symbolic program synthesis. *CoRR*, abs/1611.01855, 2016.

[120] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. Deepwalk: online learning of social representations. In *The 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '14, New York, NY, USA - August 24 - 27, 2014*, pages 701–710, 2014.

[121] Walter W Powell, Douglas R White, Kenneth W Koput, and Jason Owen-Smith. Network dynamics and field evolution: The growth of interorganizational collaboration in the life sciences. *American journal of sociology*, 110(4):1132–1205, 2005.

[122] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *CoRR*, abs/1511.06434, 2015.

[123] Sashank J Reddi, Satyen Kale, and Sanjiv Kumar. On the convergence of adam and beyond. In *International Conference on Learning Representations*, 2018.

[124] Scott Reed, Zeynep Akata, Xinchen Yan, Lajanugen Logeswaran, Bernt Schiele, and Honglak Lee. Generative adversarial text to image synthesis. In Maria Florina Balcan and Kilian Q. Weinberger, editors, *Proceedings of The 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pages 1060–1069, New York, New York, USA, 20–22 Jun 2016. PMLR.

[125] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 91–99. Curran Associates, Inc., 2015.

[126] Danilo J Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models. In *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, pages 1278–1286, 2014.

[127] Rezoolab. Make illustration on computer with chainer. http://qiita.com/rezoolab/items/5cc96b6d31153e0c86bc, 2015.

[128] Adam Roberts, Jesse Engel, Colin Raffel, Curtis Hawthorne, and Douglas Eck. A hierarchical latent vector model for learning long-term structure in music. In Jennifer Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 4364–4373, Stockholmsmssan, Stockholm Sweden, 10–15 Jul 2018. PMLR.

[129] Tim Rocktäschel, Edward Grefenstette, Karl Moritz Hermann, Tomáš Kočiskỳ, and Phil Blunsom. Reasoning about entailment with neural attention. In *International Conference on Learning Representations*, 2016.

[130] Kevin Roth, Aurelien Lucchi, Sebastian Nowozin, and Thomas Hofmann. Stabilizing training of generative adversarial networks through regularization. *arXiv preprint arXiv:1705.09367*, 2017.

[131] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3):211–252, Dec 2015.

[132] Masaki Saito and Yusuke Matsui. Illustration2vec: a semantic vector representation of illustrations. In *SIGGRAPH Asia 2015 Technical Briefs*, page 5. ACM, 2015.

[133] Lei Sha, Baobao Chang, et al. Reading and thinking: Re-read lstm unit for textual entailment recognition. In *Proceedings of the International Conference on Computational Linguistics*, 2016.

[134] Tianxiao Shen, Tao Lei, Regina Barzilay, and Tommi Jaakkola. Style transfer from non-parallel text by cross-alignment. In *Advances in Neural Information Processing Systems*, pages 6833–6844, 2017.

[135] Wenzhe Shi, Jose Caballero, Ferenc Huszár, Johannes Totz, Andrew P Aitken, Rob Bishop, Daniel Rueckert, and Zehan Wang. Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1874–1883, 2016.

[136] Stuart M Shieber. Evidence against the context-freeness of natural language. *Linguistics and Philosophy*, 1985.

[137] Xujie Si, Hanjun Dai, Mukund Raghothaman, Mayur Naik, and Le Song. Learning loop invariants for program verification. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems 31*, pages 7751–7762. Curran Associates, Inc., 2018.

[138] Xujie Si, Yuan Yang, Hanjun Dai, Mayur Naik, and Le Song. Learning a meta-solver for syntax-guided program synthesis. In *International Conference on Learning Representations*, 2019.

[139] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112, 2014.

[140] Yaniv Taigman, Adam Polyak, and Lior Wolf. Unsupervised cross-domain image generation. In *International Conference on Learning Representations*, 2017.

[141] Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. LINE: large-scale information network embedding. In *Proceedings of the 24th International Conference on World Wide Web, WWW 2015, Florence, Italy, May 18-22, 2015*, pages 1067–1077, 2015.

[142] Jie Tang, Tiancheng Lou, and Jon M. Kleinberg. Inferring social ties across heterogenous networks. In *Proceedings of the Fifth International Conference on Web Search and Web Data Mining, WSDM 2012, Seattle, WA, USA, February 8-12, 2012*, pages 743–752, 2012.

[143] Jie Tang, Jing Zhang, Limin Yao, Juanzi Li, Li Zhang, and Zhong Su. Arnetminer: extraction and mining of academic social networks. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Las Vegas, Nevada, USA, August 24-27, 2008*, pages 990–998, 2008.

[144] Wenbin Tang, Honglei Zhuang, and Jie Tang. Learning to infer social ties in large networks. In *Machine Learning and Knowledge Discovery in Databases - European Conference, ECML PKDD 2011, Athens, Greece, September 5-9, 2011, Proceedings, Part III*, pages 381–397, 2011.

[145] tdrussell. Illustrationgan. https://github.com/tdrussell/IllustrationGAN, 2016.

[146] Yingtao Tian, Haochen Chen, Bryan Perozzi, Muhao Chen, Xiaofei Sun, and Steven Skiena. Social relation inference via label propagation. In *Proceedings of the 41st European Conference on Information Retrieval, ECIR 2019, Cologe, Germany, April 14-18, 2018*, 2019.

[147] Yingtao Tian and Jesse Engel. Latent translation: Crossing modalities by bridging generative models. In *Submitted*, 2019.

[148] Chen-Tse Tsai and Dan Roth. Cross-lingual wikification using multilingual embeddings. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 589–598, 2016.

[149] Cunchao Tu, Zhengyan Zhang, Zhiyuan Liu, and Maosong Sun. Transnet: Translation-based network representation learning for social relation extraction. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI 2017, Melbourne, Australia, August 19-25, 2017*, pages 2864–2870, 2017.

[150] Shyam Upadhyay, Manaal Faruqui, Chris Dyer, and Dan Roth. Cross-lingual models of word embeddings: An empirical comparison. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 1661–1670, 2016.

[151] Aäron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew W. Senior, and Koray Kavukcuoglu. Wavenet: A generative model for raw audio. In *The 9th ISCA Speech Synthesis Workshop, Sunnyvale, CA, USA, 13-15 September 2016*, page 125, 2016.

[152] Ivan Vulić and Marie-Francine Moens. Monolingual and cross-lingual information retrieval models based on (bilingual) word embeddings. In *Proceedings of the 38th international ACM SIGIR conference on research and development in information retrieval*, pages 363–372. ACM, 2015.

[153] Yogarshi Vyas and Marine Carpuat. Sparse bilingual word representations for cross-lingual lexical entailment. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1187–1197, 2016.

[154] Fei Wang and Changshui Zhang. Label propagation through linear neighborhoods. *IEEE Transactions on Knowledge and Data Engineering*, 20(1):55–67, 2008.

[155] Jing Wang, Zhenyue Zhang, and Hongyuan Zha. Adaptive manifold learning. In *Advances in neural information processing systems*, pages 1473–1480, 2005.

[156] Ting-Chun Wang, Ming-Yu Liu, Jun-Yan Zhu, Guilin Liu, Andrew Tao, Jan Kautz, and Bryan Catanzaro. Video-to-video synthesis. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2018.

[157] Wenlu Wang, Yingtao Tian, Hongyu Xiong, Haixun Wang, and Wei-Shinn Ku. A transfer-learnable natural language interface for database. In *Submitted*, 2019.

[158] Kilian Weinberger, Anirban Dasgupta, Josh Attenberg, John Langford, and Alex Smola. Feature hashing for large scale multitask learning. *arXiv preprint arXiv:0902.2206*, 2009.

[159] David Weininger. Smiles, a chemical language and information system. 1. introduction to methodology and encoding rules. *Journal of chemical information and computer sciences*, 28(1):31–36, 1988.

[160] John Wieting, Mohit Bansal, Kevin Gimpel, and Karen Livescu. Towards universal paraphrastic sentence embeddings. *arXiv preprint arXiv:1511.08198*, 2015.

[161] Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*, 2016.

[162] Biao Xiang, Ziqi Liu, Jun Zhou, and Xiaolong Li. Feature propagation on graph: A new perspective to graph representation learning. *arXiv preprint arXiv:1804.06111*, 2018.

[163] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms, 2017.

[164] Chao Xing, Dong Wang, Chao Liu, and Yiye Lin. Normalized word embedding and orthogonal transform for bilingual word translation. In *Proceedings of the 2015 Conference of the North American Chapter*

of the Association for Computational Linguistics: Human Language Technologies, pages 1006–1011, 2015.

[165] Linchuan Xu, Xiaokai Wei, Jiannong Cao, and S Yu Philip. On exploring semantic meanings of links for embedding social networks. *Proceedings of the 27th International Conference on World Wide Web*, 2018.

[166] Wenpeng Yin and Hinrich Schütze. Convolutional neural network for paraphrase identification. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2015.

[167] Wenpeng Yin, Hinrich Schütze, et al. Abcnn: Attention-based convolutional neural network for modeling sentence pairs. *Transactions of the Association for Computational Linguistics*, 4(1), 2016.

[168] Jiaxuan You, Rex Ying, Xiang Ren, William L Hamilton, and Jure Leskovec. Graphrnn: Generating realistic graphs with deep autoregressive models. *arXiv preprint arXiv:1802.08773*, 2018.

[169] Meng Zhang, Yang Liu, Huanbo Luan, and Maosong Sun. Adversarial training for unsupervised bilingual lexicon induction. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 1959–1970, 2017.

[170] Xingxing Zhang, Liang Lu, and Mirella Lapata. Top-down tree long short-term memory networks. In *NAACL HLT 2016, The 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, San Diego California, USA, June 12-17, 2016*, pages 310–320, 2016.

[171] Xinjie Zhou, Xiaojun Wan, and Jianguo Xiao. Cross-lingual sentiment classification with bilingual document representation learning. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 1403–1412, 2016.

[172] Zhiming Zhou, Shu Rong, Han Cai, Weinan Zhang, Yong Yu, and Jun Wang. Generative adversarial nets with labeled data by activation maximization. *arXiv preprint arXiv:1703.02000*, 2017.

[173] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Computer Vision (ICCV), 2017 IEEE International Conference on*, 2017.

[174] Xiaojin Zhu and Zoubin Ghahramani. Learning from labeled and unlabeled data with label propagation. *CMU CALD tech report*, 2002.

[175] Will Y Zou, Richard Socher, Daniel Cer, and Christopher D Manning. Bilingual word embeddings for phrase-based machine translation. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1393–1398, 2013.