博士論文

A Multi-faceted Approach to Domain-aware Neural Text Generation

（ドメインを考慮したテキスト生成のための多角的な取り組み）

東京大学
THE UNIVERSITY OF TOKYO

48-177406

佐藤　翔悦

指導教員　喜連川　優

大学院　情報理工学系研究科　電子情報学専攻

東京大学

This dissertation is submitted for the degree of
Doctor of Philosophy

December 2020

Acknowledgements

I am grateful to everyone for the support during the years I have spent in the Kitsuregawa-Toyoda-Nemoto-Yoshinaga-Goda lab. First, to my supervisor, Masaru Kitsuregawa. I have enjoyed my research activity thanks to his help and the best environment for study. I will continue research activities with his words in mind, "Struggle to find a critical problem to solve." I would also like to express my gratitude to the members of my dissertation committee: Prof. Yoshimasa Tsuruoka, Prof. Sadao Kurohashi (Kyoto University), Prof. Akiko Aizawa, Prof. Koiti Hashida. Thanks to your comments and helpful pieces of advice, I could make this thesis more solid.

During my doctoral course, I have also been greatly supported by the following two professors. I would like to thank Masashi Toyoda. He spent plenty of time discussing my research while he was very busy with various tasks every day. I respect his calmness and patience. The Web data he has collected was also significantly helpful for my study. I was very fortunate to have significant support from my second supervisor, Naoki Yoshinaga. I had the most discussions and pieces of advice from him. Although I have significantly been affected by him, interestingly, I have felt philosophical differences about how NLP technology should be or will be in some parts. I will recall and miss the advice from him in the future while watching advances in NLP.

I would express my appreciation to my colleagues, other professors, and secretaries for useful discussions and helpful answers to my miscellaneous questions. Particularly to Shonosuke Ishiwatari, a senior Ph.D. student. I have learned many things from him during my five years in our lab. I am very grateful for his support, even after graduating from the lab. I want to thank Jin Sakuma, a junior Ph.D. student and a co-author of work related to this thesis. Honestly, I doubted the effectiveness of collaboration among students before, but the cooperation with him betrayed my expectation, which was a good memory for me. I would also express my appreciation to Yuma Tsuta, Amane Sugiyama, and Masato Neishi, who often gave me detailed advice on my complicated questions. I often made quick questions about

English and asked for corrections to Satoshi Tohda. I will never forget the phrase, "fall under the category." Nobukazu Fukuda was a good partner for discussion about recent papers and new ideas. I've spent a considerable amount of time working on personalization studies with Daisuke Oba as a co-author. Thanks to his efforts, I was able to deepen my thoughts on areas that I could not study on my own due to lack of time.

Finally, I would like to dedicate this thesis to my parents, who have supported my life.

Abstract

Machine translation, dialogue, automatic summarization, and story generation – technologies for text generation are deeply related to the important NLP tasks. Researchers have eagerly explored methods for text generation with high quality. The large diversity of probable outputs causes major difficulties in text generation. Compared to tasks with only one correct answer, the patterns of probable outputs (i.e., sequences of words) are countless. Moreover, the criterion for evaluating outputs depends significantly on the goal, the situation, and the domain where text generation is needed. For this reason, employing text generation for practical use has often required developers to spend high costs of designing and manually customizing a system for each purpose.

Recently, the rapid increase in data on the Web and advances in machine learning represented by neural networks have motivated researchers to adopt data-driven approaches for text generation. Although the approaches allowed developers to reduce the costs notably, it also raised another problem; the performance of data-driven models significantly drops in specialized domains that are different from training data. However, sufficient in-domain data for training is not necessarily available. As a result, in the age of data-driven approaches, developers have difficulty in collecting a large amount of in-domain data instead of manually designing an in-domain system.

Many practitioners have explored methods to handle the difference in domains to resolve the problem, also known as domain adaptation. Assuming a situation where the amount of in-domain data is small, they exploit a large amount of out-domain data with methods such as data-selection, fine-tuning, and multi-domain learning. In the existing studies, the difference in domains is often treated as the difference in datasets. This tradition is due to the convenience of experiments and the high costs of annotating fine-grained domain tags to data.

Although such attempts have had some success, it is still unclear which factors in domain differences can affect models' performance. We consider there is room for

improvement by exploiting fine-grained domains in a multi-faceted approach. Our challenges and approaches are threefold:

- **Vocabulary adaptation**: Domain differences can affect the vocabulary that appears in the text and its meaning. However, due to the scarcity of in-domain parallel data available for training, differences in vocabulary and meaning of words were difficult problems to solve by the existing domain adaptation methods. In particular, in a model based on a neural network that has been frequently used in recent years, the vocabulary is often built in the phase of pre-processing. The scheme has hindered us from applying domain adaptation methods to embedding layers of models. We propose a method to directly adapt the embedding layers of a trained model by using the embeddings pre-trained from in-domain monolingual corpora.

- **Situation-aware generation**: Examples in a dataset for text generation are not made under the same situation even if they are in the same dataset; who, when, where, and under what circumstances an example was created can all potentially affect generation, particularly in tasks where outputs can have high diversity. We attempt to exploit the situations in dialogue with conversation data collected from social media.

- **Speculative sampling of latent variables**: The difficulty in handling fine-grained domains in data is mainly attributed to the cost of labeling or collection. Domains that can affect generated outputs are not necessarily available; they may not be publicly available due to privacy concerns or not recorded as a label. Such inaccessible domains can also involve what output a model should generate. We try to capture such inevitable randomness in text generation by latent variables of variational models and propose a method for solving the problem that makes it difficult to train models that can generate diverse outputs.

Table of contents

List of figures

List of tables

Chapter 1

Introduction

1.1 Data-driven Text Generation

In the history of natural language processing (NLP), teaching machines to generate high-quality texts has been one of the most important goals. This is not only because text generation technologies are useful but also because the ability to express one's thoughts or choice as a text – the ability of communication – was considered as one of the forms of intelligence [111, 110].

Early approaches mainly relied on rule-based, template-based, or IR-based methods. The considerable advantage of these approaches is the controllability of outputs. By manually defining rules, filtering, and alignments between inputs and outputs, developers can avoid the risk of generating low-quality outputs. However, the obvious problem is that probable outputs, sequences of words, are countless in text generation. This requires developers to spend high costs to prepare a sufficient amount of patterns for achieving high-quality generation. Thus, the research trend shifted to data-driven text generation with the rapid increase of data on the Web [92].

As with many NLP methods in recent years, data-driven text generation has eagerly been studied in machine translation and has spread to other tasks. In the age of statistical machine translation (SMT) [11], the methods for SMT were applied to dialogue modeling [92]. In recent years, the encoder-decoder framework, a standard method of neural-based machine translation (NMT), is an epoch-making development and fascinated NLP researchers. Although it was originally proposed for machine translation, nowadays, it is employed as a common tool even for other generation tasks [106, 113, 125, 88, 77, 76].

One of the considerable advantages of the method is its versatility; In the approach, text generation is modeled as a one-to-one projection on a shared framework among tasks, which allows us to quickly transfer promising techniques proposed in an NLP task to another task. Thus, we employ neural-based text generation methods as core technologies in this thesis and propose task-agnostic methods to improve existing methods.

Despite the enormous efforts of our predecessors, however, it is hard to say that current neural-based text generation satisfies humans' requirements. It is still in the middle of being put into practical use except in several tasks such as machine translation. We will discuss the issues of current approaches and our challenges in Section 1.2.

1.2 Research Challenges

In common settings, models are trained and tested under the closed-world assumption [90] – the domains of training and testing are the same. The training data is also expected to contain a sufficient number of examples. The assumption often does not hold due to the large costs of data collection. Practically, in many cases, it is inevitable to employ a model trained from out-domain training data. The performance of such a model tends to drop substantially, as reported by Koehn and Knowles [49]. Therefore, many existing studies to address the problem, collectively referred to as **domain adaptation**, have tried to handle the domains of data [19, 67, 43, 48, 115, 109, 55, 17].

The important problems we focus on in this thesis are as follows: 1) domain specificity of words and meaning, and 2) domain specificity of input-output correspondence. Related to the first problem, methods for handling unknown words in general domains have been eargerly explored for machine translation, separately from the efforts for domain adaptation [39, 64, 98, 51, 20]. The major recent approaches were based on dictionaries, copying, and subwords (or characters). However, such approaches are inadequate to cover the problems caused by domains. It is costly to prepare a large domain-specific dictionary or a knowledge-base for each domain. It is not necessarily possible to infer the meaning from the surface of a word because domain-specific terms are proper nouns in many cases (e.g., "alsa/pulseaudi", the name of a package in the computer domain). Additionally, for text generation tasks such as non-factoid question answering or dialogue modeling in specialized domains, copying or transliteration can be inapplicable since it is probable that such domain-specific terms appear only in either the input or the output. Thus, it is essential to develop a method for handling the word-level domain specifity even in data-scarse and specialized domains.

Next, we discuss the second problem. Although we mentioned earlier that text generation is typically modeled as a one-to-one mapping from input to output, the relation between inputs and outputs is practically *many-to-many* due to the fine-grained domains of data. Even in machine translation, where the output is likely to be constrained by the input text strictly, the style, the length, and the word choices can be diverse. For example, in En→Ja translation, a pronoun "you" can be translated into "あなた," "君,", and "お前." The suitable choice from probable outputs is affected by many factors; when assuming a formal situation, "あなた" would be an appropriate choice. In a conversation with a friend with whom you can feel at ease, "君" can be used. We generally build a dataset by collecting examples made under similar situations and treat the whole dataset as a coarse-grained domain. However, even among examples in the same dataset, there exist implicit differences in fine-grained domains (e.g., writer, time, etc.), and they can affect the input-output correspondences. We collectively refer to these factors as domains, regardless of whether or not the factors are explicitly given to data as labels. Failing to handle the diversity in data can skew the training of models and lead them to a generation of too bland and less informative outputs [53].

Conversely, the utilization of domains is helpful in diversifying and controlling system outputs. In the tasks such as dialogue response generation or story generation, the high diversity of system outputs is naturally preferred to keep users attracted. Even for tasks such as machine translation or grammatical error correction, where the diversity of outputs is often not experimentally evaluated, it would be useful in practical use to display various probable output candidates for users and would help non-native writers write sophisticated sentences depending on their purpose.

We aim to handle the aforementioned domain-specific input-output correspondence with two approaches in this thesis. In the first approach, we assume that domains of each example in a dataset are explicitly available as discrete labels and explore methods for effectively utilizing them. In the second approach, we assume that fine-grained domain differences implicitly exist even in the same dataset and aim to model them as randomness.

We here list overviews of our challenges:

1. **Handling domain-specific words and meanings**: Domain differences can affect the vocabulary that appears in the text and its meaning. However, due to the scarcity of in-domain parallel data available for training, differences in vocabulary and meaning of words were difficult problems to solve by the existing domain adaptation methods. In particular, a neural-based model's vocabularies are often built in the phase of pre-processing without paying attention to the difference in domains. This manner hinders us from applying domain adaptation methods to the meaning of words or subwords.

2. **Explicit modeling of domain-specific input-output correspondence**: Examples in a dataset for text generation are not made under the same condition even if they are in the same dataset; who, when, where, and under what circumstances an example was created can all potentially affect the result, particularly in tasks where outputs can have high diversity. We collectively treat such fine-grained domains as situations. Conventional models without taking situations into consideration tend to be uninformative and cannot adjust the outputs to users' preferences.

3. **Implicit modeling of domain-specific input-output correspondence**: The difficulty in handling fine-grained domains in data is mainly attributed to the cost of labeling or collection. Domains that can affect generated outputs are not necessarily available; they may not be publicly available due to privacy concerns or not recorded as a discrete label. Such inaccessible domains can also involve what output a model should generate. Variational models are

a promising approach to implicitly capture latent and inaccessible domains in text generation. However, the training of variational models is unstable, and it often happens that diversification of outputs is not achieved.

We target two major text generation tasks in this thesis – machine translation and dialogue modeling. Although both of the two tasks are major and expected to be put into practical use, their characteristics and problems are different.

The first approach will be evaluated on machine translation, and the second and third approaches will be evaluated on dialogue modeling. The reason why we selected machine translation for the first approach is the ease of analysis in the task. In translation, the alignments between input and output allow us to confirm how domain-specific words are translated by a model. On the contrary, in text generation tasks where the outputs are diverse, the safe response problem makes it difficult to perform detailed analysis focusing on vocabulary and word meanings.

We selected dialogue modeling for the second and third approaches because the approaches are proposed for handling the diversity of outputs. In dialogue, responses to an utterance can be affected by a wide variety of factors than other generation tasks such as machine translation and automatic summarization, which is suitable for confirming the effects of our proposed method to consider fine-grained domains and diversify outputs.

In the following section, we will explain our approaches corresponding to the three challenges and the contribution in this thesis.

1.3 Contribution

In this thesis, we work on the three following approaches to address the challenges described in Section 1.2.

1. **Vocabulary adaptation**: To handle domain-specific words and meanings, we propose vocabulary adaptation, an inexpensive method to directly adapt the embedding layers of a trained model by using the embeddings pre-trained from in-domain monolingual corpora. While conventional neural-based models assume static vocabularies constructed before applying

domain adaptation, our proposed method enables us to change them and provides more suitable vocabularies depending on the domain. The evaluation results on machine translation showed that our method improved the performance of models by 3.28 – 3.86 BLEU score in situations where a large amount of in-domain parallel data is not available.

2. **Situation-aware generation**: To explicitly model fine-grained domains that can affect text generation, we attempt to exploit the situations of conversation data in dialogue modeling. Concretely, we propose two models: 1) local-global SEQ2SEQ and 2) SEQ2SEQ with situation embeddings assuming that situations are available as discrete labels. The response selection tests on a massive amount of Twitter datasets confirmed the effectiveness of using situations.

3. **Speculative sampling of latent variables** We try to handle implicit subdomains in a dataset based on variational models. To resolve the problem in conventional variational models, we proposed speculative sampling that samples multiple latent variables from the posterior distribution of a model and chooses the most probable one for modulating the latent space. The results of automatic and human evaluation confirmed that our method mitigated KL vanishing, and generated outputs were specific while keeping relevance to contexts.

For achieving the practical use of text generation, domain-aware methodologies are indispensable because: 1) users naturally expect a system to process domain-specific terms, and 2) the ability to generate diverse outputs and the customizability depending on the situation motivate users to continue using the system. We expect that our attempts in this thesis will help the future activities of domain-aware text generation.

1.4 Thesis Structure

This thesis is structured as follows. In Chapter 2, we first explain preliminary knowledge of core technologies that our approaches are based on. The following three chapters, we present the details of our approaches to the challenges. In

Chapter 3, we focus on the difference of vocabulary and meanings of words between domains and propose a method for directly adapt the embedding layers of an NMT model. In Chapter 4, we propose a method of introducing situations of conversations to a dialogue model. In Chapter 5, we aim to resolve KL vanishing, an important problem of training variational text generation models that are promising approaches to handling implicit and fine-grained domains. Finally, we present a conclusion of our work in Chapter 6.

Chapter 2

Preliminary Knowledge

In this chapter, we introduce preliminary knowledge of the methods our studies are based on, mainly neural networks employed for NLP. Although many types of neural networks have been employed for NLP, we here describe the overview of feed-forward neural network (FNN), word embeddings [71], recurrent neural network (RNN) [22], and RNN-based encoder-decoder model [15, 106] attention-mechanism [4], and Transformer [112] as core architectures of our systems.

2.1 Basis of Neural Network

We first explain how an input text is processed through a neural-based model while taking a feed-forward neural network (FFN) for classification as an example. We here suppose that the input is a sentence (= a list of words), and the output is a probability distribution of output classes. In text generation, the output is a sequence of probability distributions. The output classes correspond to each word in the vocabulary. About neural networks for text generation, we will discuss in Section 2.2.

One of the simplest form of FFN, also known as multi-layered perceptron (MLP) consists of three layers: an input layer (a.k.a. embedding layer) $\mathbf{X} = \{\mathbf{x_0}, \mathbf{x_1}, \cdots, \mathbf{x_{T_x-1}}\}$, hidden layers $\mathbf{H} = \{\mathbf{h_0}, \mathbf{h_1}, \cdots, \mathbf{h_n}\}$, and an output layer $\mathbf{y}$. We here denote a sequence of input words by $X = \{x_0, x_1, \cdots, x_{T_x-1}\}$ and their vector representations (a.k.a. word embeddings) as $\mathbf{X} = \{\mathbf{x_0}, \mathbf{x_1}, \cdots, \mathbf{x_{T_x-1}}\}$. T_x means the length of inputs and n is the number of hidden layers.

In the standard use of neural networks for NLP, input features (i.e., a list of words) are predefined word-IDs represented as one-hot vectors. The input layer transforms the inputs into a list of continuous vectors $\mathbf{X}$. The continuous vectors, called *word embeddings*, are the representations of words in the feature space of the network. It then computes the sentence-level representation by summarizing the continuous vector sequence $\mathbf{h_0}$ (by an operation such as averaging).

The subsequent non-linear transformation is iteratively applied to the previous hidden layers $\mathbf{h_i}$. Finally, the last hidden layer is transformed into a probability distribution $\mathbf{y}$.

The whole process of the model is formulated as follows:

$$\mathbf{h_0} = avg\left(\mathbf{W_x}\mathbf{X}\right) \tag{2.1}$$

$$\mathbf{h_{i+1}} = f_h\left(\mathbf{W_{h_i}}\mathbf{h_i} + \mathbf{b_{h_i}}\right) \tag{2.2}$$

$$\mathbf{y} = f_y\left(\mathbf{W_y}\mathbf{h_n} + \mathbf{b_y}\right) \tag{2.3}$$

Here, $\mathbf{W}_*$ and $\mathbf{b}_*$ are trainable parameters of the model. T_x and n denote the number of input words and hidden layers, respectively. $f(\cdot)$ is an activation function for introducing non-linearity to the model.

For the activation function of the hidden layers f_h, sigmoid function:

$$f(\mathbf{z})_j = \frac{1}{1+e^{-z_j}}, \tag{2.4}$$

hyperbolic tangent:

$$f(\mathbf{z})_j = tanh(\mathbf{z})_j = \frac{e^{z_j} - e^{-z_j}}{e^{z_j} + e^{-z_j}}, \tag{2.5}$$

and rectified linear unit (ReLU):

$$f(\mathbf{z})_j = max(0, z_j) \tag{2.6}$$

are commonly employed. In the formulas above, $f(\mathbf{z})_j$ denotes the the jth dimension value of the vector $\mathbf{z}$ that is input to the activation function.

Typically, the softmax function is employed as f_y that is defined by the formula:

$$f(\mathbf{z})_j = \frac{e^{z_j}}{\sum_{k=1}^{K} e^{z_k}} \tag{2.7}$$

This function is used for normalizing the transformed last hidden layer into a probability distribution. K is the number of output classes.

2.2 RNN-based Encoder-Decoder Model

Recurrent Neural Network

The weak point of the architecture we described in Section 2.1 is that it cannot handle the order of input words when inducing a sentence-level representation. Furthermore, when taking the order of words into consideration (e.g., by assigning different parameters to each word position), the maximum length of inputs has to be fixed. For this reason, the simple FFN is not suitable to be used for data represented as a sequence such as a text. Employing Recurrent Neural Network (RNN) has been considered as one of the promising approaches for such data.

There exist many variants of RNN, including modern architectures such as Long-short Term Memory (LSTM) [35] and Gated Recurrent Unit (GRU) [15]. We here introduce Elman network [22] as the simplest form of RNN while assuming that it takes a sequence of words as an input. Recurrent neural network is a model that has a directed acyclic graph inside. Concretely, it takes an input word embedding $\mathbf{x_t}$ and the previous output (i.e., hidden layer) $\mathbf{h_{t-1}}$ at each time step t, and computes the next output $\mathbf{h_{t-1}}$.

The computation of RNN is defined as:

$$\begin{aligned} \mathbf{h_t} &= f_h\,(\mathbf{W_h}\,\mathbf{x_t} + \mathbf{U_h}\,\mathbf{h_{t-1}} + \mathbf{b_h}) \\ \mathbf{y_t} &= f_y\,(\mathbf{W_y}\,\mathbf{h_t} + \mathbf{b_y}) \end{aligned} \tag{2.8}$$

$\mathbf{W}_*$, $\mathbf{U}_*$, and $\mathbf{b}_*$ are trainable parameters of the model. x_t is an embedding of the word input at the time step t. The state at $t = 0$ is randomly initialized or fixed to a trainable vector learned through the same process as other trainable parameters.

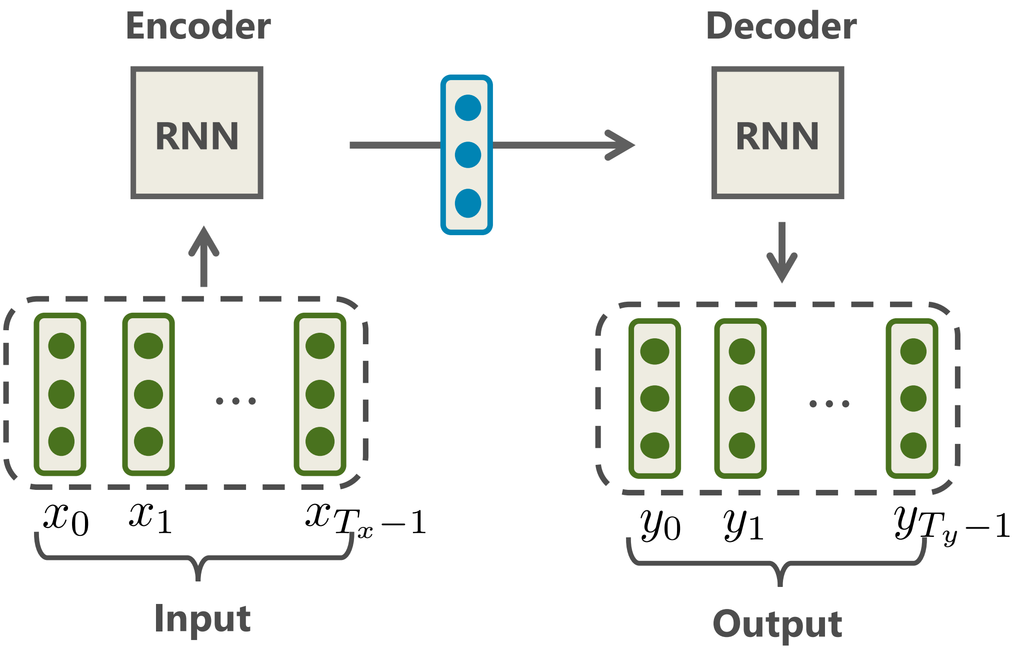

Figure 2.1: Overview of encoder-decoder model: illustrative example.

As the formula shows, the length of outputs from RNN $\{\mathbf{h_0}, \mathbf{h_1}, \cdots, \mathbf{h_{T_x-1}}\}$ is equal to the length of the input words $\{\mathbf{x_0}, \mathbf{x_1}, \cdots, \mathbf{x_{T_x-1}}\}$ when the length of inputs is m. For tasks such as language modeling [70, 65], where a model needs to predict the output word for each input word, y_t is the output corresponding to x_t. For tasks where a model requires a sentence-level representation to generate outputs such as sentence classification and text generation, the hidden layers are summarized by taking the average/maximum, or the last hidden layer is used as the representation of the sentence [15, 102, 101, 41]. These operations are called average-/max-/last-pooling, respectively. For example, using only y_{T_x-1} corresponds to last-pooling in Eq. (2.8).

Encoder-Decoder Model

The encoder-decoder, also known as SEQ2SEQ is a model for text generation that consists of two RNNs [15, 106]. As we discussed in the previous part, RNN can encode a sentence at an arbitrary length while considering the order of words in the sentence (Figure 2.1).

One naive approach to using RNN for text generation is to select each of the most probable words from the sequence of probability distributions $\mathbf{y} = \{\mathbf{y_0}, \mathbf{y_1}, \cdots, \mathbf{y_{T_x-1}}\}$ in Eq. (2.8). However, there are drawbacks in this approach; the length of output has to be identical to the input words. Besides, each of the output words is **independently** sampled from the distribution without knowing which word a model sampled in the previous time step.

To solve these problems, encoder-decoder models have another RNN for iteratively generating the output sentence. The RNN is called decoder-RNN, while the RNN for reading an input sentence is called encoder-RNN. As the architecture of the encoder-decoder model is the combination of two RNNs, the computation is similar to Eq. (2.8).

Specifically, the encoder-RNN first computes the hidden states of an input sentence by encoding each input word. The hidden state at the last time step is then fed to decoder-RNN as the initial hidden state.

2.3 Attention Mechanism

After the emergence of the encoder-decoder model, many studies have reported promising performance in NLP tasks. However, the architecture assumed that the length of the input and the output are relatively short. When handling long-term dependency, the performance tended to drop. The problem was mainly attributed to the iterative computations in conventional RNNs; they update the hidden state for each time step, and tokens appearing at the beginning of an input sentence tend to be forgotten.

To resolve the problem, *attention mechanism* was proposed [4, 63] for paying attention to each token depending on the model's state instead of using the last hidden state of the encoder. The key idea of the attention mechanism is to dynamically weight multiple vectors (i.e., encoder's hidden states) by using a state vector (i.e., a decoder's hidden state) as a query vector. Suppose that we have a query vector $\mathbf{q}$ and N pairs of key-value vectors $\mathbf{K} = \{\mathbf{k_0}, \mathbf{k_1}, \cdots, \mathbf{k_{N-1}}\}$ and $\mathbf{V} = \{\mathbf{v_0}, \mathbf{v_1}, \cdots, \mathbf{v_{N-1}}\}$, the computation is defined as:

$$\begin{aligned} \mathbf{c} &= \sum_{i=0}^{N-1} \alpha_i \mathbf{v_i} \\ e_i &= \mathbf{q}^{\mathrm{T}} \cdot \mathbf{k_i} \\ \alpha_i &= \frac{\exp(e_i)}{\sum_{i'=1}^{N-1} \exp(e_{i'})} \end{aligned} \tag{2.9}$$

where **c** is the output of the computation by the attention mechanism. In standard RNN-based NMT models, **K** and **V** denote outputs from an encoder. q denotes a hidden state of a decoder. The computation of e_i we introduced in Eq. (2.9) is called multiplicative (or dot-product) attention. Meanwhile, the other commonly used computation is called additive attention, where e_i is computed as:

$$e_i = \mathbf{a}^{\mathrm{T}} \cdot \tanh(\mathbf{Wq} + \mathbf{Uk_i} + \mathbf{b}) \tag{2.10}$$

a, **W**, **U**, and **b** are trainable parameters of the model.

2.4 Transformer

In this section, we describe an overview of Transformer [112], a recent encoder-decoder model that fully exploits the attention mechanism. Transformer is a de facto standard model for text-to-text generation. As the overall structure of Transformer is complicated, we only mention the major differences compared to the RNN-based encoder-decoder model.

The important difference of Transformer from traditional RNN-based encoder-decoder models is the computation of sentence-level representations. It does not rely on recursive computations for encoding a sequence of vector representations. Instead, it aggregates the sequence by the attention-mechanism called *self-attention*. The computation by self-attention on the i th encoder layer is as follows. Suppose that there is a sequence of representation $\mathbf{H^i} = \{\mathbf{h^i_0}, \mathbf{h^i_1}, \cdots, \mathbf{h^i_{N-1}}\}$ as inputs to the layer. The layer computes the outputs $\mathbf{H^{i+1}}$ fed to the next layer. Here, to compute

$\mathbf{h}_{\mathbf{j}}^{\mathbf{i+1}}$ following Eq. (2.9), $\mathbf{h}_{\mathbf{j}}^{\mathbf{i}}$ is used as a query vector, and $\mathbf{H}^{\mathbf{i}}$ is used as key-value vectors.

Also, in the decoder, the computation of self-attention is the same as the encoder. However, a decoder layer has no access to the representations $\{\mathbf{h}_{\mathbf{j+1}}^{\mathbf{i}}, \mathbf{h}_{\mathbf{j+2}}^{\mathbf{i}}, \cdots, \mathbf{h}_{\mathbf{N}}^{\mathbf{i}}\}$ when computing $\mathbf{h}_{\mathbf{j}}^{\mathbf{i+1}}$ because they correspond to future tokens to be generated in the decoder. Thus, the tokens are masked when applying self-attention; the attention weight α is fixed to zero in Eq. (2.9).

We should also mention that the self-attention-based computation is not aware of the positions of tokens differently from iterative encoders such as RNN. To complement them, standard Transformers employ positional embeddings. Transformer concatenates word (or subword) embeddings with additional embeddings that represent the feature of each position.

Chapter 3

Vocabulary Adaptation for Neural Machine Translation

3.1 Introduction

The performance of neural machine translation (NMT) models remarkably drops in domains different from the training data. Since a massive amount of parallel data is available only in a limited number of domains, domain adaptation is often required to employ NMT in practical applications. Researchers have therefore developed fine-tuning, a dominant approach for this problem [61, 24, 16, 107, 42, 8] (Section 3.2). Assuming a massive amount of out-domain and small amount of in-domain parallel data, fine-tuning adjusts the parameters of a out-domain pre-trained model.

However, in fine-tuning, inheriting the embedding layers of the out-domain pre-trained model causes vocabulary mismatches; namely, a model can handle neither domain-specific words that are not covered by a small amount of in-domain parallel data (*unknown words*) nor words that have different meanings across domains (*semantic shift*). Moreover, adopting the standard subword tokenization [98, 51] accelerates the semantic shift. domain-specific words are often finely decomposed into out-domain subwords (*e.g.*, "alloy" → "_all" + "o" + "y"), which introduces improper subword meanings and hinders adaptation (Table 3.8 in Section 3.5).

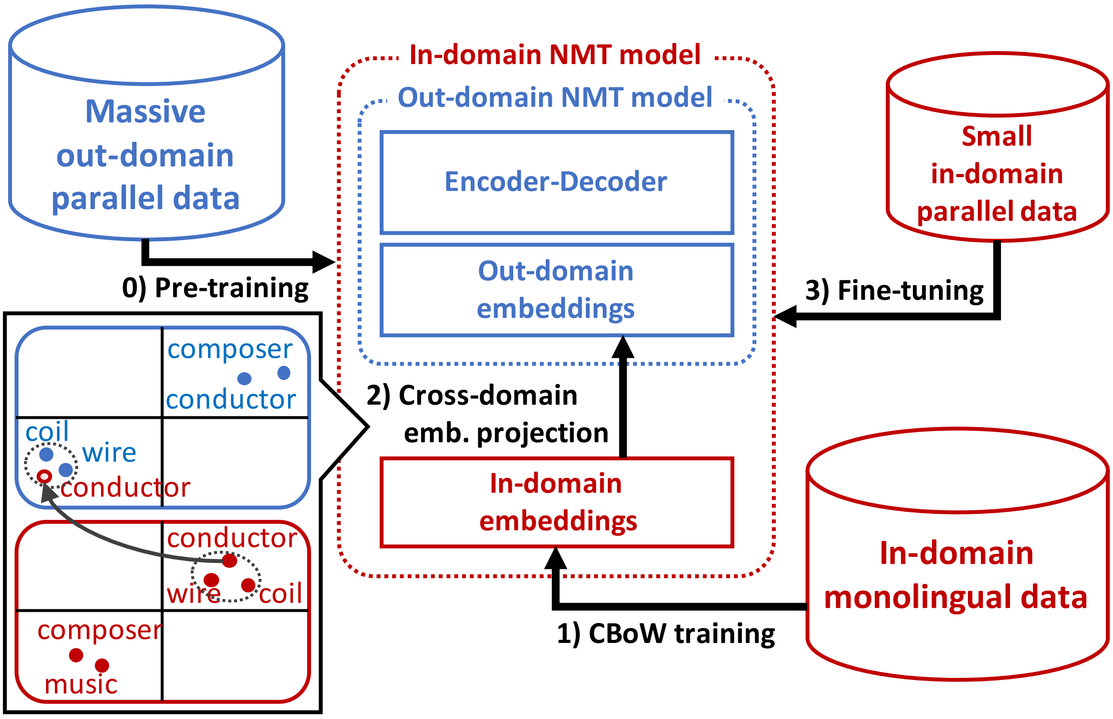

Figure 3.1: Vocabulary adaptation for domain adaptation in NMT using cross-domain embedding projection.

To resolve these vocabulary-mismatch problems in domain adaptation, we propose *vocabulary adaptation* (Figure 3.1), a method of directly adapting the vocabulary (and embedding layers) of a pre-trained NMT model, to perform effective fine-tuning (Section 3.3). Given an out-domain pre-trained NMT, we first induce a wide coverage of in-domain word embeddings from in-domain monolingual data. We then fit the obtained in-domain word embeddings to the embedding space of the pre-trained NMT model by inducing a cross-domain projection from the in-domain embedding space to the out-domain embedding space. To perform this cross-domain and cross-task embedding projection, we explore two methods: cross-lingual [123] and cross-task embedding projection [93].

We evaluate fine-tuning with the proposed vocabulary adaptation for two domain pairs: 1) from **JESC** [86] to **ASPEC** [74] for English to Japanese translation (En→Ja) and 2) from the **IT** domain to **Law** domain [49] for German to English translation (De→En). Experimental results demonstrate that our vocabulary adaptation improves the BLEU scores [84] of fine-tuning [61] by 3.86 points (21.45

to 25.31) for En→Ja and 3.28 points (24.59 to 27.87) for De→En (Section 3.5). Moreover, it shows further improvements when combined with back-translation [97].

The contributions of this work are as follows.

- We empirically confirmed that **vocabulary mismatches hindered domain adaptation**.
- We established **an effective, model-free fine-tuning for NMT** that adapts the vocabulary of a pre-trained model.
- We showed that **vocabulary adaptation exhibited additive improvements over back-translation** that uses monolingual corpora.

3.2 Related Work

In this section, we first review two approaches to supervised domain adaptation in NMT: multi-domain learning and fine-tuning. We then introduce unsupervised domain adaptation using in-domain monolingual data and approaches to unknown word problems in NMT.

Multi-domain learning induces an NMT model from parallel data in both domains [48, 115, 10]. Since this approach requires training with a massive amount of out-domain parallel data, the training cost becomes problematic when we perform adaptation to many domains.

Fine-tuning (or continued learning) is a standard domain adaptation method in NMT. Given an NMT model pre-trained with a massive amount of out-domain parallel data, it continues the training of this pre-trained model with a small amount of in-domain parallel data [61, 16, 107, 8, 30]. Due to the small cost of training, research trends have shifted to fine-tuning from multi-domain learning. Recent studies focus on model architectures, training objectives, and strategies in training. Meanwhile, no attempts have been made to resolve the vocabulary mismatch problem in domain adaptation.

Unsupervised domain adaptation exploits in-domain monolingual data to train a language model to support the model's decoder for generating natural in-domain sentences [32, 21]. Data augmentation using back-translation [97, 37] is another approach to using in-domain monolingual data.

These approaches can partly address the problem of semantic shift. However, it is possible that the out-domain encoder will fail to handle domain-specific words. In such cases, a decoder with the in-domain language model becomes less helpful in the former approach, and the generated pseudo-parallel corpus has low-quality sentences on the encoder side in the latter approach.

Handling unknown words has been extensively studied for NMT since the vocabulary size of an NMT model is limited due to practical requirements (*e.g.*, GPU memory) [39, 64]. The current standard approach to the unknown word problem is to use token units shorter than words such as characters [56, 62] and subwords [98, 51] to handle rare words as a sequence of known tokens. However, more drastic semantic shifts will occur for characters or subwords than for words because they are shorter than words and naturally ambiguous.

Besides these studies mentioned above, we introduce related work that appeared after this study was published as Sato et al. [94]. Aji et al. [2] reported that transferring embeddings and vocabulary mismatches between parent and child models significantly affected the performance of models also in cross-lingual transfer learning. The idea of Poerner et al. [85] is quite similar to our approach although their method was for adapting pre-trained language models to NER and QA tasks.

3.3 Vocabulary Adaptation for Domain Adaptation in NMT

As we have discussed (Section 3.1), the vocabulary mismatch problem between domains is the important challenge in domain adaptation for NMT. This section proposes fine-tuning-based methods of directly resolving this problem. Although our methods are applicable to any NMT model with embedding layers, we assume here subword-based encoder-decoder models [4, 112] for clarity.

3.3.1 Vocabulary Adaptation Prior to Pre-training

One simple approach is to use in-domain vocabularies in pre-training. Specifically, we first construct vocabularies from *in-domain* data for each language. We then pretrain an out-domain NMT model with the in-domain vocabularies and embeddings. Finally, we fine-tune the pre-trained model with in-domain parallel data.

In this approach, however, employing the in-domain vocabularies will hinder out-domain pre-training. In addition, since the embeddings induced from the in-domain data are tuned to the domain of pre-training, the problem of semantic shifts still remains and will hinder fine-tuning.

3.3.2 Vocabulary Adaptation Prior to Fine-tuning

Another approach is to replace the encoder's embeddings and the decoder's embeddings of the pre-trained NMT model with word embeddings induced from in-domain data before fine-tuning. However, as in transplanting organs from a donor to a recipient, this causes rejection; the embedding space of a pre-trained model is irrelevant to the space of the in-domain word embeddings.

We therefore project the in-domain word embeddings onto the embedding space of the pre-trained model in order to make the embeddings compatible with the pre-trained model (Figure 3.1 in Section 3.1). This approach is inspired by cross-lingual and cross-task word embeddings that bridge word embeddings across languages and tasks.

An overview of our proposed method is given as follows.

Step 1 (Inducing in-domain embeddings) We induce word embeddings from in-domain monolingual data for each language. Although we can use any method for induction, we adopt Continuous Bag-of-Words (CBOW) [71] here since CBOW is effective for initializing embeddings in NMT [75], which suggests embedding spaces of CBOW and NMT are topologically similar.

Step 2 (Projecting embeddings across domains) We project the in-domain embeddings of the source and target languages into the embedding spaces of the pre-trained encoder and decoder, respectively, to obtain cross-domain embeddings.

Step 3 (Fine-tuning) We replace the vocabularies and the embedding layers with the cross-domain embeddings and apply fine-tuning using the in-domain parallel data.

To induce cross-domain embedding projection, we regard the two domains as different languages/tasks and explore the use of methods for inducing cross-lingual [123] and cross-task word embeddings [93]. In what follows, we explain each method.

Vocabulary Adaptation by Linear Transformation

The first method exploits an orthogonal linear transformation [123] to obtain cross-lingual word embeddings. We use subwords shared across two domains for inducing an orthogonal linear transformation from the in-domain embedding space to the out-domain embedding space. The obtained linear transformation is used to map all in-domain embeddings to the out-domain embedding space to address semantic shift across domains.

Concretely, suppose that we have an embedding x_i to be projected, a target embedding z_i, and trainable parameters W, the projection method maximizes the cosine similarity between the projected embedding Wx_i and the target embedding z_i as follows:

$$\max_{W} \sum_{i} (Wx_i)^T z_i. \tag{3.1}$$

After updating the parameters for projection W, W is orthogonalized by solving the following constrained quadratic problem:

$$\min_{\bar{W}} ||W - \bar{W}|| \text{ s.t. } \bar{W}^T \bar{W} = I. \tag{3.2}$$

Vocabulary Adaptation by Locally Linear Mapping

Due to the difference between the domains and tasks (CBOW and NMT) in inducing the embeddings, the linear transformation is likely to fail. Thus, we employ a recent method for cross-task embedding projection called "locally linear mapping" (LLM) [93]. An overview is illustrated in Figure 3.1 (lower left).

LLM learns a projection that preserves the local topology (positional relationships) of the original embeddings after mapping while disregarding the global topology. This property of LLM is suited to our situation because the local topology is expected to be the same across the semantic spaces of two domains, while globally, they can be significantly different due to semantic shift between domains as illustrated in Figure 3.2.

Here, we explain the essence of LLM. Interested readers may consult Sakuma and Yoshinaga [93] for details. Suppose that T^{LM} is the in-domain word embeddings induced by a language model task, and S^{NMT} is the out-domain word embeddings induced by the translation task (the embedding layer of the pre-trained model). We denote the vocabulary of T^{LM} by V_T, the vocabulary of S^{NMT} by V_S and the vocabulary of words shared across both domains by $V_{\mathrm{shared}} = V_T \cap V_S$.

Our goal is to produce embeddings T^{NMT} with a vocabulary of V_T in the embedding space of S^{NMT}. We accomplish this by computing the T^{NMT} that best preserves the local topology of T^{LM} in the embedding space of S^{NMT}. Concretely, for each word w_i in V_T, we first take the k-nearest neighbors $\mathcal{N}(w_i) \subset V_{\mathrm{shared}}$ in T^{LM}. We use cosine similarity as the metric for the nearest neighbor search.

Second, we learn the local topology around w_i by reconstructing $T^{\mathrm{LM}}_{w_i}$ from the embeddings of its nearest neighbors as a weighted average. For this purpose, we minimize the following objective:

$$\hat{\boldsymbol{\alpha}}_i = \operatorname*{argmin}_{\boldsymbol{\alpha}_i} \left\| T^{\mathrm{LM}}_{w_i} - \sum_{w_j \in \mathcal{N}(w_i)} \alpha_{ij} T^{\mathrm{LM}}_{w_j} \right\|^2 , \tag{3.3}$$

with the constraint of $\sum_j \alpha_{ij} = 1$; the method of Lagrange multipliers gives the analytical solution.

We then compute the embedding $T^{\mathrm{NMT}}_{w_i}$ that preserves the local topology by minimizing the following objective function:

$$T^{\mathrm{NMT}} = \operatorname*{argmin}_{T^{\mathrm{NMT}}} \left\| T^{\mathrm{NMT}}_{w_i} - \sum_{w_j \in \mathcal{N}(w_i)} \hat{\alpha}_{ij} S^{\mathrm{NMT}}_{w_j} \right\|^2 . \tag{3.4}$$

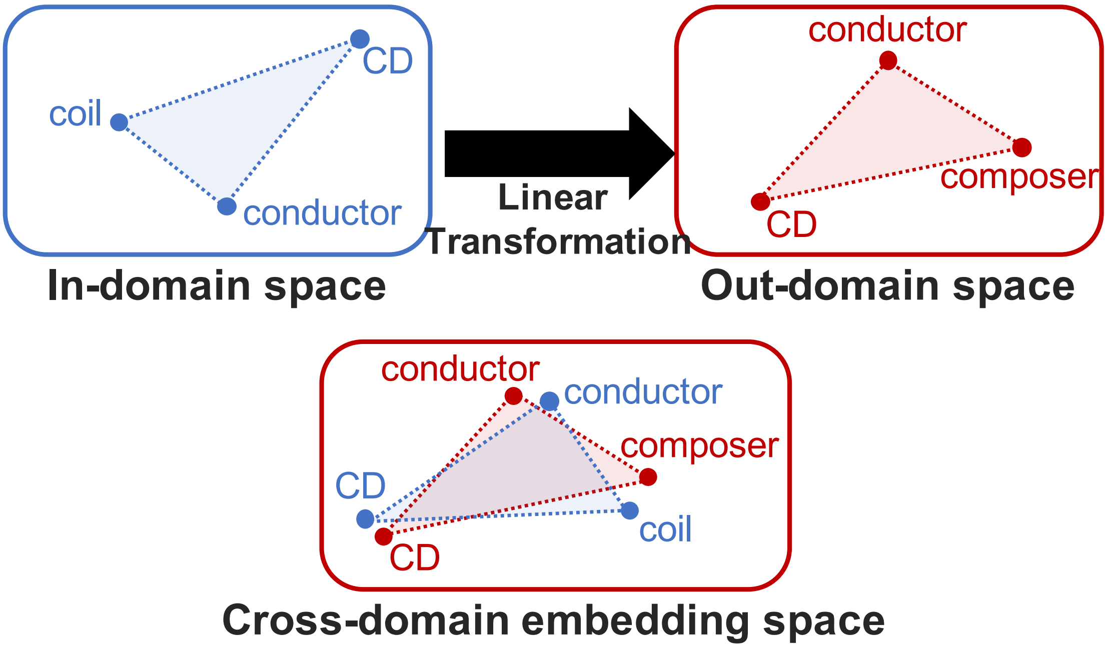

Figure 3.2: **Unwanted** cross-domain projection by linear transformation due to difference of topology in vector-based embedding space: illustrative example.

This optimization problem has the trivial solution:

$$T_{w_i}^{\text{NMT}} = \sum_{w_j \in \mathcal{N}(w_i)} \hat{\alpha}_{ij} S_{w_j}^{\text{NMT}}. \tag{3.5}$$

Note that subwords shared across domains will have different embeddings after projection ($T_w^{\text{NMT}} \neq S_w^{\text{NMT}}$ for $w \in V_{\text{shared}}$). This captures the semantic shift of subwords across domains. We conduct a detailed analysis of this matter in Section 3.6.3.

3.4 Experimental Setup

We conducted fine-tuning with our vocabulary adaptation for domain adaptation in En→Ja and De→En machine translation. In what follows, we describe the setup of our experiments.

3.4.1 Datasets and Preprocessing

We selected domain pairs to simulate a plausible situation where in-domain data is specialized and similar out-domain parallel data is not available.

For En→Ja translation, we chose the Japanese-English Subtitle Corpus (JESC) [86][1] as out-domain data and Asian Scientific Paper Excerpt Corpus (ASPEC) [74][2] as in-domain data. JESC was constructed from subtitles of movies and TV shows, while ASPEC was constructed from abstracts of scientific papers. These domains are substantially distant, and ASPEC contains many technical terms that are unknown in the JESC domain. We followed the official splitting of training, development, and test sets, except that the last 1,000,000 sentence pairs were omitted in the training set of the ASPEC corpus as they contain low-quality translations.

For De→En translation, we adopted the dataset constructed by Koehn and Knowles [49] from the OPUS corpus [108]. This dataset includes multiple domains that are distant from each other and is suitable for experiments on realistic domain adaptation. We chose the **IT** domain and the **Law** domain from the dataset as the out-domain and in-domain data, respectively. We followed the same splitting of training, development, and test sets as Koehn and Knowles [49].

Preprocessing As preprocessing for the En→Ja datasets, we first tokenized the parallel data using the Moses toolkit (v4.0)[3] for English sentences and KyTea (v0.4.2)[4] for Japanese sentences. We then truecased the English sentences by using the script in the Moses toolkit. As for the De→En datasets, we used the same tokenization and truecasing as Koehn and Knowles [49]. The statistics of the datasets are listed in Table 3.1.

We applied SentencePiece (v0.1.83)[5] [52] trained from the monolingual data in each domain to the tokenized datasets. The number of subwords was 16,000 for all languages. In the training of SentencePiece, we did not concatenate the input

[1] https://nlp.stanford.edu/projects/jesc/

[2] http://orchid.kuee.kyoto-u.ac.jp/ASPEC/

[3] https://github.com/moses-smt/mosesdecoder

[4] http://www.phontron.com/kytea

[5] https://github.com/google/sentencepiece

En→Ja		**JESC**	**→**	**ASPEC**
# examples	training (all)	2,797,388		2,000,000
	development	2,000		1,790
	testing	-		1,812
# distinct words (En)		161,695		637,377
# distinct words (Ja)		169,649		384,077
# shared words (En)		46,950 (7.4% in ASPEC)		
# shared words (Ja)		50,003 (13.0% in ASPEC)		
De→En		**IT**	**→**	**Law (Acquis)**
# examples	training (all)	337,817		715,372
	development	2,526		2,000
	testing	-		2,000
# distinct words (De)		140,508		189,084
# distinct words (En)		70,650		92,316
# shared words (De)		21,912 (11.6 % in Law)		
# shared words (En)		17,165 (18.6 % in Law)		

Table 3.1: Statistics of out-domain and in-domain parallel corpus. #distinct/shared words are counted in training sets.

language and output language to maximize the portability of the pre-trained model.

From each of the preprocessed datasets, we used 1) 100,000 randomly sampled sentence pairs or 2) all sentence pairs in the training set for in-domain training. This was for evaluating models in both cases where we have a small/large in-domain dataset.

To prepare reproducible in-domain monolingual data, we shuffled and divided all sentence pairs of the in-domain training set except the 100,000 sentence pairs into two equal portions. We then used the first half and the second half as simulated monolingual data for the source language and the target language, respectively.

The monolingual data was used for training in-domain SentencePiece, CBOW vectors, and data augmentation by back-translation. When models did not use the monolingual data, the data used for training SentencePiece and CBOW vectors was exactly identical to the training set in each domain.

# encoder/decoder layers	6	Label smoothing rate	0.1
# attention heads	8	Init. learning rate	1e-3
Dim. of embeddings	512	(warmup)	1e-7
Dim. of Transformer	2048	Dropout rate	0.1
Vocab. size (enc&dec)	16k	Beam size for decoding	5
Max. tokens in batch	64k	Length penalty	1.2

Table 3.2: Hyperparameters of NMT models.

3.4.2 Models and Embeddings

We adopted Transformer-base [112] implemented in fairseq (v0.8.0)[6] [83], as the core architecture for the NMT models.[7] Major hyperparameters are shown in Table 3.2.[8] We evaluated the performance of the models on the basis of BLEU [84]. Before pre-training the models, we induced subword embeddings from the monolingual corpus by Continuous Bag-of-Words (CBOW) [71] to initialize the embedding layers of the NMT models.

To evaluate the effect of vocabulary adaptation, we compared the following settings (and their combinations) that used either or both the out- and in-domain parallel data.

Out-/In-domain trains a model only from the training set in the out-/in- domain.

Fine-tuning w/ out-domain vocab. (FT-outV) continues to train the **Out-domain** model using the in-domain training set without any vocabulary adaptation [61].

Fine-tuning w/ in-domain vocab. (FT-inV) Refer to Section 3.3.1.

Multi-domain learning (MDL) trains a model from both out- and in- domain training sets. We employed domain token mixing [10] as a method of multi-domain learning. In this setting, we jointly used the training sets of both domains

[6] https://github.com/pytorch/fairseq

[7] Note that since Transformer shares the embedding and output layers of the decoder, vocabulary adaptation is applied to the embedding layer of the encoder and the tied embedding/output layer of the decoder, respectively.

[8] For De→En translation, we made minor modifications to the architecture to follow Hu et al. [37]. Concretely, we added layer normalization [3] before each of the encoder and decoder stacks. We also applied dropout to the outputs of the activation functions and self-attention layers.

for training subword tokenization models, CBOW vectors, and training NMT models (e.g., 2797k + 100k for En→Ja translation). This method prepends a special token of the current domain (*e.g.*, `<src>`) to the target sentence in training. This enforces the decoder to predict the current domain from the input, which works as regularization.

Vocabulary Adaptation (VA) Refer to Section 3.3.2. We compared two projection methods: linear orthogonal transformation (**VA-Linear**) and locally linear mapping (**VA-LLM**) described in Section 3.3.2. For **VA-LLM**, the number of nearest neighbors, *k*, was fixed to 10.[9] To highlight the importance of embedding projection for the proposed method, we also evaluated settings using the in-domain CBOW vectors for the re-initialization as is (VA-CBoW).

Back-translation (BT) applies a backward translation to in-domain monolingual data in the target language. We employed the most standard back-translation proposed by Sennrich et al. [97]. For this back-translation, a backward model (*e.g.*, Ja → En) is independently trained from the out-domain parallel data with the same setting and data as **Out-domain**. The subsequent fine-tuning is applied with the generated pseudo-parallel in-domain corpora and a in-domain training set.

Among the above methods, **Out-domain** and **In-domain** do not perform domain adaptation. **FT-outV**, **FT-inV**, and **MDL** are baseline domain adaptation methods. **BT** is applied to **FT-outV**, **FT-inV**, and **VA** for data augmentation.

Note that **FT-inV** and **MDL** assume that the target domain is given before training with the out-domain data. Although this assumption enables us to build in-domain suitable vocabularies, it sacrifices the domain portability of trained models. As a result, it requires us to perform training for a long period of each combination of domains.

We used Adam [44] to train each model with the above settings. During both pre-training and fine-tuning, the learning rate linearly increased for warm-up for the first 4,000 training steps and then decayed proportionally to the inverse square root of the number of updates. Prior to fine-tuning, we reset the optimizer

[9] We tested **VA-LLM** with k={1, 5, 10, 20} in preliminary experiments, and the default value (k=10) was the best.

	# In-domain data			
	En → Ja		**De → En**	
	100k	2000k	100k	715k
No adaptation				
Out-domain	4.61		2.58	
In-domain	11.69	41.83	18.79	34.16
Baselines				
MDL	21.65	41.92	24.03	37.74
FT-outV	21.45	43.09	24.59	38.43
FT-inV	**28.08**	42.32	24.87	36.38
Proposed				
VA-CBoW	15.28	41.44	21.88	36.34
VA-Linear	22.26	42.70	25.20	37.00
VA-LLM	21.79	**43.96**	**26.40**	**39.41**

Table 3.3: Case-sensitive BLEU scores for NMT domain adaptation: En→Ja from **JESC** to **ASPEC** and De→En from **IT** to **Law**. Size of training set for **Out-domain** was 2797k for **JESC** and 338k for **Law**.

	Enc	**Dec**	**En → Ja**		**De → En**	
			100k	2000k	100k	715k
FT-outV			21.45	43.09	24.59	38.43
	✓		**22.69**	43.48	25.64	39.48
VA-LLM		✓	20.75	43.66	25.69	**40.19**
	✓	✓	21.79	**43.96**	**26.40**	39.41

Table 3.4: BLEU scores on ablation tests for **VA-LLM**.

and the learning rate and then continued training on the in-domain training set. For vocabulary adaptation, we replaced the embedding layers of the out-domain pre-trained models with the projected in-domain word embeddings except for the four special tokens (<pad>, <unk>, </s>, <s>). Both in training and fine-tuning, we saved checkpoints at the end of epochs, and we adopted the model at the checkpoints with the best validation loss on the development set.

3.5 Results

3.5.1 BLEU Scores

Table 3.3 shows the results for the domain adaptations. Among all the methods, **VA-LLM** achieved the best BLEU score in three out of the four cases. The low BLEU scores for **Out-domain** show how much domain mismatch degraded the NMT performance, as pointed out in [49]. There were large differences in the performance among **VA-*** models that perform vocabulary adaptation prior to fine tuning. The results confirmed that not only the differences in the vocabulary (set of subwords) but also the initial embeddings matter in fine-tuning NMT models.

VA-* methods did not work well in En→Ja translation when only the 100k in-domain parallel data was used. This is probably because the more noisy emebeddings (ambiguous subwords) introduced by the large number of domain-specific words in the ASPEC dataset (Table 3.1) hinders the embedding projection of **VA-LLM** and **VA-Linear** with low-quality CBOW vectors trained from the 100k sentences. In this setting, we need more parallel data for fine-tuning to adjust the noisy initial embeddings.

Table 3.4 shows results of ablation tests to examine for which side (encoder or decoder) **VA-LLM** benefited. The results confirmed that the poor performance in En→Ja translation with the 100k in-domain parallel data is due to the failure of handling semantic shifts in the decoder.[10]

The improvements obtained by **VA-Linear** were modest overall. This was due to the nature of the linear projection employed for cross-domain embedding mapping as discussed in Section 3.3.2. We analyze the difference between the two types of projected embeddings in Section 3.6.3.

3.5.2 Effects of Monolingual Data

Table 3.5 shows how employing in-domain monolingual data affected domain adaptation. In the settings, the in-domain SentencePiece and CBOW vectors were trained from both the 100k parallel data and the monolingual data (950k and 308k

[10] We observed the same tendency when we conducted the ablation tests for Ja→En translation with the ASPEC datasets.

	# In-domain data			
	En → Ja		**De → En**	
	100k	**+BT**	100k	**+BT**
FT-outV	21.45	24.63	24.59	25.81
w/ monolingual data for training CBoW				
FT-inV	18.85	21.75	21.87	24.49
VA-Linear	19.35	22.19	24.09	25.79
VA-LLM	**25.31**	**29.73**	**27.87**	**28.43**

Table 3.5: Case-sensitive BLEU scores when employing target-domain monolingual data (950k for En→Ja and 308k for De→En). +BT indicates that monolingual data was used also for data augmentation.

for En→Ja and De→En, respectively). We also evaluated the orthogonality of the proposed method to **BT** since both methods exploit in-domain monolingual data.

Interestingly, the results of **FT-inV** and **VA-Linear** were worse than the results in Table 3.3. We consider the reason to be as follows. When additionally using the in-domain monolingual data, the resulting in-domain SentencePiece model and CBOW vectors become more suitable thanks to the increase of data. However, this also means that domain-specific words appearing only in the monolingual data accelerated the vocabulary mismatches, the semantic shifts, and the difference of topology in the embedding space. As the result, the vocabulary mismatches degraded the out-domain pre-trained model for **FT-inV** and linear transformation failed to handle the semantic shifts for **VA-Linear**.

In contrast, due to the capability of the projection method, the performance of **VA-LLM** was successfully improved by the use of the monolingual data. Table 3.5 also shows the orthogonality of **VA-LLM** to **BT**, since the increase of BLEU scores for **VA-LLM + BT** from **FT-outV + BT** were substantial (5.10 pt and 2.61 pt for En→Ja and De→En translation, respectively).

We should mention that the proposed model did not work well and generated ungrammatical outputs in fully unsupervised situations without fine-tuning. The model was largely degraded and its BLEU score for En→Ja translation was 0.01, almost zero. It is not surprising because in such a situation the out-domain model was required to utilize the in-domain vocabulary and the cross-domain embeddings that had been optimized for different training objectives. The degree of the degradation could depend on the noises caused by the embedding

	En → Ja	De → En
FT-outV	12.19	10.79
VA-LLM	12.94	11.19

Table 3.6: Case-sensitive BLEU scores when using only out-domain parallel data and pseudo parallel data generated by back-translation.

projection and the sensitivity of the out-domain NMT model to the change of paramters. Ideally, although the embedding projections resolve the difference in the embedding space, noises could remain and drastically degrade the model. The combination with back-translation or employing small in-domain parallel data for fine-tuning is a solution for mitigating the problem as shown in the results above. However, vocabulary adaptation only by embedding projections is fascinating considering the convenience. As future work, we will analyze how the noises degrade pre-trained models before fine-tuning and improve the methods for embedding projection.

Furthermore, Table 3.6 shows that how our method performed in a extreme setting where no in-domain parallel data is available. Concretely, in the fine-tuning step, we retrained **Out-domain** with only the pseudo-parallel data generated by back-translation. In this setting, in-domain CBOW vectors were also trained only from in-domain monolingual data (i.e., the number of in-domain sentences was changed from 1050k to 950k for En→Ja). However, the quality of cross-domain embeddings was similar to those in Table 3.5. Although **VA-LLM** performed better than **FT-outV**, the improvements were modest. We consider there were two reasons. First, when no in-domain parallel data was available and the overall performance of a baseline model was poor, it even failed to handle frequent phrases, and thus the capability of processing domain-specific terms was less influential. Second, as the in-domain data used for fine-tuning was generated by back-translation using out-domain NMT model. Sentences on the source-language side did not contain in-domain terms and thus the embeddings of transferred in-domain subwords were infrequently updated through fine-tuning.

	# Updates in training w/		
	source	**target**	**BT**
	(2797k)	(100k)	(950k)
w/o monolingual data			
In-domain	-	3,440	-
MDL	36,342		-
FT-outV	28,750	2,480	-
VA-LLM	28,750	3,200	-
w/ monolingual data			
FT-outV + BT	56,350	31,280	
VA-LLM + BT	56,350	32,895	

Table 3.7: Number of updates until convergence for En→Ja translation.

3.5.3 On Efficiency: Training Steps

Table 3.7 shows the number of updates until convergence in En→Ja translation with the 100,000 in-domain training set.[11] We confirmed that all models were trained over a sufficient number of steps. The validation loss did not improve over at least five epochs after the best model was chosen. We used four GPUs (NVIDIA Quadro P6000) for training, and it took 0.9 sec/update on average.

Here, we emphasize that **VA-LLM** achieved superior performance with a small number of updates (3,200 steps, less than 50 minutes) similarly to **FT-outV**. Note that the overhead time of our vocabulary adaptation was negligible since embedding projection took only several minutes. Meanwhile, **FT-outV + BT** took 31,280 steps due to the size of the augmented data even when we ignore the time taken to generate back-translated parallel data.

Additionally, our proposed method is based on fine-tuning and the target domain is not supposed to be given before out-domain pre-training, differently from **MDL**. Therefore, the pre-trained **Out-domain** can be reused each time when the target domain or settings are changed, which enables us to omit the long training time (28,750 steps, about 7.2 hours) per model training. As the training steps of **VA-LLM + BT** show, the overhead caused by employing the proposed method with back-translation was also small. Nevertheless, the improvements of **VA-LLM + BT** compared with **FT-outV + BT** were substantial (Table 3.5).

[11] As for **FT-outV + BT** and **VA-LLM + BT**, the number of updates in the pre-training phase is the sum of the training steps for both forward and backward models.

Input (JESC vocab.)	_the _function _of _server _was _strengthen ed _in _order _to _strengthen _the **_I nt ra net**$_1$ ś _mechanism _.
Input (ASPEC vocab.)	_the _function _of _server _was _strengthened _in _order _to _strengthen _the **_Intra net**$_1$ ś _mechanism _.
Reference	イ ン ト ラ ネ ッ ト の し く み を 強化 する ため に サーバー 機能 の 強化 を 行 っ た 。
FT-outV	**Imagenet** の 機構 を 強化 する ため に , サーバ の 機能 を 強化 し た 。
FT-outV + BT	サーバ の 機能 を 強化 する ため に , **Intelligent** の 機能 を 強化 し た 。
VA-LLM + BT	サーバ 機能 は イ ン ト ラ ネ ッ ト$_1$ の 機能 を 強化 する ため に 強化 さ れ た 。
Input (JESC vocab.)	_3 _cases _of _the _lu m bar _spinal$_1$ _can al **_ste no s is**$_2$ ⋯
Input (ASPEC vocab.)	_3 _cases _of _the _lumbar _spinal$_1$ _canal **_stenosis**$_2$ ⋯
Reference	⋯ 腰部 脊柱$_1$ 管 狭窄 症$_2$ の 3 例 に つ い て ⋯
FT-outV	⋯ 腰部 <unk> 柱 管 狭 <unk> 症 の 3 症例 に つ い て ⋯
FT-outV + BT	⋯ 腰部 <unk> 柱 管 狭 <unk> 症 の 3 症例 に つ い て ⋯
VA-LLM + BT	⋯ 腰部 脊柱$_1$ 管 狭窄$_2$ の 3 症例 に つ い て ⋯
Input (IT vocab.)	_falls _der _Austausch _der **_Rat if ik ation s ur ku nden**$_1$ _zwischen ⋯
Input (Law vocab.)	_falls _der _Austausch _der **_Ratifikation surkunde n**$_1$ _zwischen ⋯
Reference	should the instruments of **ratification**$_1$ be exchanged between ⋯
FT-outV	if the exchange of the **ratification** of **ratification** between ⋯
FT-outV + BT	where the exchange of the Council takes place between ⋯
VA-LLM + BT	if the instruments of **ratification**$_1$ are met between ⋯

Table 3.8: Translation examples of the models with 100k target-domain parallel data in Table 3.3 and Table 3.5. **Bolded words** are rare or unknown when pretraining. Underlined words and subscript numbers indicate correspondence. Input (JESC, IT) and Input (ASPEC, Law) were fed to **FT-outV/FT-outV + BT** and **VA-LLM + BT**, respectively.

3.6 Analysis

3.6.1 Translation Examples

Table 3.8 shows translation examples generated by **FT-outV** in Table 3.3, **FT-outV + BT** and **VA-LLM + BT** in Table 3.5. The size of in-domain parallel data for training was 100k.

FT-outV and **FT-outV + BT** often failed to translate in-domain words that were tokenized into short subwords. In such cases, the models tended to ignore or transliterate them. For instance, the De→En examples (lower) show that **FT-outV** and **FT-outV + BT** failed in translating "Ratifikationsurkunden (instruments of ratification)."

Moreover, in the En→Ja examples (upper), the decomposed domain-specific words "脊柱 (spinal)" and "狭窄症 (stenosis)" contained domain-specific subwords such as "脊" and "窄." The models without vocabulary adaptation also failed to handle these subwords when both the out-domain training set and the in-domain 100k training set rarely contained them.

Meanwhile, **VA-LLM + BT** successfully translated both of the cases with the help of in-domain monolingual data. These examples imply the difficulty in translating domain-specific words without vocabulary adaptation.

We observed that the outputs generated by**VA-LLM + BT** contained various domain-specific words. To quantitatively confirm this, we calculated the percentage of distinct words included in both the generated outputs and the references. The outputs in En→Ja translation generated by **VA-LLM + BT**, **FT-outV + BT**, and **FT-outV** contained 57.9%, 53.4%, and 49.5% of distinct words in the references, respectively.

3.6.2 Effect of Vocabulary Size in Fine Tuning

As reported in [99], the vocabulary size of an NMT model can affect its translation quality in a low-resource setting. How about in fine-tuning? To explore this, we varied only the **in-domain** vocabulary size of **VA-LLM** before fine-tuning by vocabulary adaptation.

Figure 3.3 shows that **VA-LLM** preferred large vocabulary sizes when additional in-domain monolingual data was used for training CBOW, whereas it

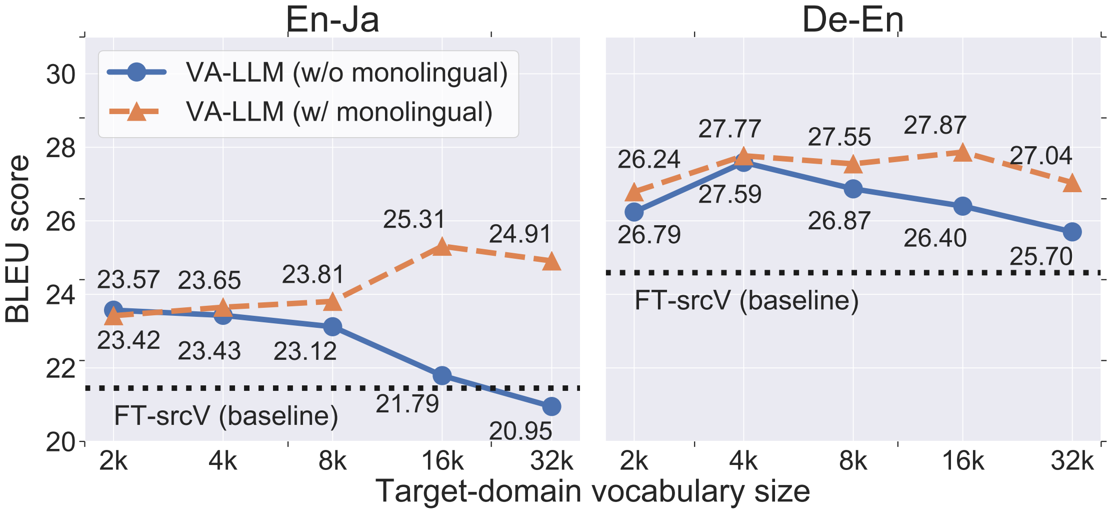

Figure 3.3: BLEU scores of VA-LLM while varying in-domain vocabulary size. The out-domain vocabulary size was fixed to 16k.

preferred small vocabulary sizes when the data was not used. We consider the reason to be as follows. In the former case, a large vocabulary contains low-frequency subwords of which representation is unlikely to be well-trained as discussed in [99]. In the latter case, however, in-domain monolingual data can cover such low-frequency subwords.

As this analysis showed, the vocabulary size also had large effects on fine-tuning (3.52 pt difference at most). Besides the vocabulary mismatch problem, our vocabulary adaptation could make further improvements by the vocabulary size were adjusted depending on the amount of in-domain parallel and monolingual data with a low training cost.

3.6.3 Quality of Cross-domain Embeddings

The advantage of our approach is that it adjusts the meanings of subwords (embeddings) as well as the vocabulary (set of subwords) to the target domain. We thus examined to what extent our vocabulary adaptation captures the semantic shift.

Nearest neighbors in ASPEC-cbow embedding space	
_branches	_branch, _roots, _veins, _arteries, _trees
_experimentally	_systematically, _numerical, _theoretical, _experimental, _experiments
Nearest neighbors in JESC-nmt embedding space	
via linear transformation (Linear)	
_branches	**_trees**, _sides, _birds, _parts, _pieces
_experimentally	_rope, _tanks, _laser, _gravitational, _simulation
via locally linear mapping (LLM)	
_branches	**_branch**, **_trees**, **_roots**, **_veins**, **_arteries**
_experimentally	_by, _experiment, **_experiments**, **_experimental**, _simulation

Table 3.9: Top-5 nearest neighbors of *“_branches”* and *“_experimentally”* in ASPEC-cbow embedding space and JESC-nmt embedding space via cross-domain embedding projection: **bold-faced** subwords are nearest neighbors shared across both top-5. The ASPEC-cbow vectors are trained from the 100k target-domain parallel data and the monolingual data.

We first observed the nearest neighbors based on cosine similarity for each of the in-domain subword embeddings (hereafter, ASPEC-cbow).[12] Note that the nearest neighbors should be unchanged even after embedding projection to keep the in-domain meanings.

Next, we compute cosine similarities between each of the projected ASPEC-cbow and the embeddings of **Out-domain** to find their nearest neighbors in the embedding space of **Out-domain** (hereafter, JESC-nmt). The obtained nearest neighbors show how the ASPEC-cbow embeddings projected by linear-transformation or LLM performed during fine-tuning.

Table 3.9 shows the nearest neighbors of two words: *“_branches,”* which appears in both domains and can have different meanings across domains, and *“_experimentally,”* which is only in the ASPEC domain.

While the cbow vector for *“_branches”* and the embedding projected by LLM have the meaning of “_veins” and “_arteries”, the embedding projected by linear transformation lost it. *“_experimentally”* is a subword that only the in-domain

[12] Through this analysis, the candidates of nearest neighbors were limited to the shared subwords across JESC and ASPEC domains for clear comparison.

(ASPEC) vocabulary contains. As illustrated in Figure 3.2, the mapping of domain-specific subword embeddings is likely to fail due to the difference of topology in the embedding space. We found that LLM relatively accurately computed its embedding in the JESC-NMT space while linear transformation failed. This tendency was also observed when using only the 100k parallel data for training of SentencePiece and CBOW vectors. These observations demonstrate the capability of LLM in cross-task/domain embedding projection.

3.7 Chapter Summary

In this study, we tackled the crux of the vocabulary mismatch problem in domain adaptation for NMT, and we proposed vocabulary adaptation, a simple but direct solution to this problem. It adapts the vocabulary of a pre-trained NMT model for performing effective fine-tuning. Regarding domains as independent languages/tasks, our method makes wide-coverage word embeddings induced from in-domain monolingual data be compatible with a out-domain pre-trained model.

We explored two methods for projecting word embeddings across two domains: linear transformation and locally linear mapping (LLM). The experimental results for English to Japanese translation and German to English translation confirmed that our domain adaptation method with LLM dramatically improved the translation performance. We will release all code to promote the reproducibility of our results.[13]

Although the vocabulary adaptation was evaluated only for NMT, where it was easy to conduct solid analyses, it is also applicable to a wider range of neural network models and tasks, and it can even be combined with existing fine-tuning-based domain adaptations. In other text generation tasks, the problem of differences in vocabularies and meanings is the same or can be larger. In machine translation, copying or transliteration without understanding the meaning of words can be one of the possible methods to handle infrequent terms. For example, when translating a sentence “Why is alsa/pulseaudi not working?”, models do not necesarily the meaning of “alsa/pulseaudi.” However, for example, in tasks

[13] https://github.com/jack-and-rozz/vocabulary_adaptation

such as question answering or dialogue modeling, users often require agents to be an expert of a field and a good assistant that can not only superficially process but also understand such infrequent terms; otherwise, the agents will only say “I am sorry, I do not understand alsa/plseaudi.” to the question. We leave empirically confirming the effect of our method in other text generation tasks as future work where the diversity of outputs is high and thus it is difficult to conduct an evaluation.

Chapter 4

Situation-Aware Dialogue

4.1 Introduction

The increasing amount of dialogue data in social media has opened the door to data-driven modeling of non-task-oriented, or chat, dialogues [92]. The data-driven models assume a response generation as a sequence to sequence mapping task, and recent ones are based on neural SEQ2SEQ models [114, 102, 53, 54, 124]. However, the adequacy of responses generated by these neural models is somewhat insufficient, in contrast to the acknowledged success of the neural SEQ2SEQ models in machine translation [40].

The contrasting outcomes in machine translation and chat dialogue modeling can be explained by the difference in the degrees of freedom on output for a given input. An appropriate response to a given utterance is not monolithic in chat dialogue. Nevertheless, since only one ground truth response is provided in the actual dialogue data, the supervised systems will hesitate when choosing from the vast range of possible responses.

So, how do humans decide how to respond? We converse with others while (implicitly) considering not only the utterance but also other various conversational situations (Section 4.3) such as time, place, and the current context of conversation and even our relationship with the listener. For example, when a friend says "I feel so sleepy." in the morning, a probable response could be "Were you up all night?" (Figure 4.1). If the friend says the same thing at midnight, you

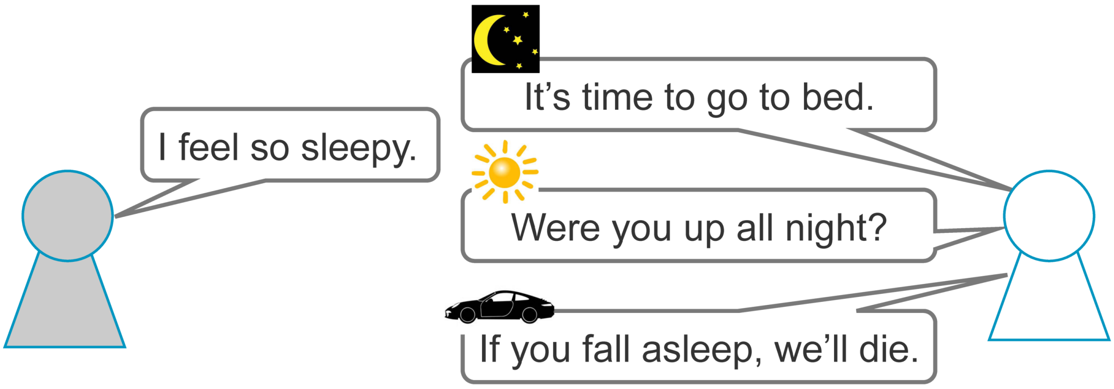

Figure 4.1: Conversational situations and responses.

might say “It’s time to go to bed.” Or if the friend is driving a car with you, you might answer “If you fall asleep, we’ll die.”

Modeling situations behind conversations has been an open problem in chat dialogue modeling, and this difficulty has partly forced us to focus on task-oriented dialogue systems [119], the response of which has a low degree of freedom thanks to domain and goal specificity. Although a few studies have tried to exploit conversational situations such as speakers’ emotions [34] or personal characteristics [54] and topics [124], the methods are specially designed for and evaluated using specific types of situations.

In this study, we explore neural conversational models that have general mechanisms to incorporate various types of situations behind chat conversations (Section 4.4.2). These models take into account situations on the speaker’s side and the listener’s side (or those who respond) when encoding utterances and decoding its responses, respectively. To capture the conversational situations, we design two mechanisms that differ in how strong of an effect a given situation has on generating responses.

In experiments, we examined the proposed conversational models by incorporating three types of concrete conversational situations (Section 4.3): utterance, speaker/listener (profiles), and time (season), respectively. Although the models are capable of generating responses, we evaluate the models with a response selection test to avoid known issues in automatic evaluation metrics of generated responses [57]. Experimental results obtained using massive dialogue data from

Twitter showed that modeling conversational situations improved the relevance of responses (Section 4.5).

4.2 Related Work

Conversational situations have been implicitly addressed by preparing datasets specific to the target situations and by solving the problem as a task-oriented conversation task [119]; examples include troubleshooting [114], navigation [117], interviewing [47], and restaurant search [118]. In what follows, we introduce non-task-oriented conversational models that explicitly consider conversational situations.

Hasegawa et al. [34] presented a conversational model that generates a response so that it elicits a certain emotion (*e.g.*, joy) in the listener mind. Their model is based on statistical machine translation and linearly interpolates two conversational models that are trained from a small emotion-labeled dialogue corpus and a large non-labeled dialogue corpus, respectively. This model is similar to our local-global Seq2Seq but differs in that it has hyperparameters for the interpolation, whereas our local-global Seq2Seq automatically learns $\boldsymbol{W}_G$ and $\boldsymbol{W}_L$ from the training data.

Li et al. [54] proposed a neural conversational model that generates responses taking into consideration speakers' personalities such as gender or living place. Their model cannot handle unknown speakers, because it feeds a specific speaker ID to their model and represent individual (known) speakers with embeddings. In contrast, our model can consider any speakers with profiles because we represent each cluster of profiles with an embedding and find an appropriate profile type for the given profile by nearest-neighbor search.

Sordoni et al. [105] encoded a given utterance and the past dialogue exchanges, and combined the resulting representations for rnn to decode a response. Zhao et al. [129] used a conditional variational autoencoder and automatically induced dialogue acts to handle discourse-level diversity in the encoder.

Xing et al. [124] proposed to explicitly consider topics of utterances to generate topic-coherent responses. They used latent Dirichlet allocation while we use k-means clustering. Both methods confirmed the importance of utterance situations.

The way to obtain specific situations is still an open research problem. As demonstrated in this study, our primary contribution is the invention of neural mechanisms that can consider various conversational situations.

Our local-global Seq2Seq model is closely related to a many-to-many multi-task Seq2Seq proposed by Luong et al. [60]. The critical difference is in that their model assumes only local tasks, while our model assumes many local tasks (situation-specific dialogue modeling) and one global task (general dialogue modeling).

Hereafter, we also review related work that appeared after this study was published [96]. First, we describe studies of personalization that have been eargely explored due to its importance. The study of Zhang et al. [128] is closely related to our study. The most standard method proposed by Li et al. [54] relied on past conversations of each spearker, and thus it was difficult to handle unknown speaker's persona. For the reason, Zhang et al. [128] also employed a profile (description) of speakers and directly encoded for inducing the speaker's embeddings. Mazare et al. [68] constructed a large dataset for personalization from reddit data. Their method can be regarded as the combination of Li et al. [54] and Zhang et al. [128]. Chu et al. [18] proposed a character trope classification task to handle personas from dialogues. While we aim to obtain such character tropes by clustering (i.e., speaker profile clusters), their model learned them through the classification task by using the CMU Movie Summary dataset [7]. Bak and Oh [6] used a speaker's persona for inducing distributions in a variational model.

Overall, the persona of speakers has been regarded an important factor to model a conversation and many methods for exploiting personas have been developed. However, the problem of how to collect persona information still remains; although most of the studies relied on past utterances or his/her profile to infer a speaker's persona, such clues are not necesarily available and do not represent all of their preferences, thoughts, and demographic factors. As a future direction, we consider that it is still necessary to explore methods for utilizing more various types of factors for inferring personas as we aimed in this study.

Second, we discuss the relation to task-oriented dialogue modeling. In task-oriented dialogue modeling, one of the main focuses is on dialogue state tracking (DST), where models fill in predefined slots with values that are extracted or estimated from conversations (e.g., {*price=cheap, location=New York*}). Traditional approaches prepared a dataset specific to a single situation and it was a difficult challenge to handle multiple situations because predefined slots and probable values were inevitably dependent on the domain.

Recently, the emergence of a large-scale multi-domain dataset, MultiWOZ [12], motivated researchers to tackle multi-domain dialogue state tracking [89, 121, 26, 91, 127, 13]. However, the domains targeted by these approaches were coarse-grained and the aim was different from that of non-task-oriented dialogue modeling. Specifically, the main difficulty in multi-domain DST appeared as the difference in probable slot-value pairs between domains.

As an important difference, we should mention that the response to a dialogue state tended to be predefined in task-oriented dialogue modeling. This is because the main aims of task-oriented dialogue systems are to work as the interface of services (e.g., information retrieval, booking, and the like). Thus, the diversity of outputs is not required there and it can be not costly to manually prepare responses or templates for each dialogue state type. Recently, neural-based response generation in task-oriented dialogue has also begun to be explored [12, 14]. In the method of Budzianowski et al. [12], extracted dialogue states are fed to models as jointly trained embeddings similarly to Li et al. [54] and Sato et al. [96]. In more advanced approaches, attention-based methods were employed to represent dialogue states and dialogue acts as hierarchical graph for response generation [14].

Conversely, is it possible to use a dialog state as a helpful clue even in non-task-oriented dialogue? For such a use, the problem is that predefined dialogue states in current conventional DST are typically domain-specific (e.g., price range for hotel booking) while non-task-oriented dialogues contain various domains. Thus, we consider it is an important but difficult issue how to design dialogue states and state trackers that can be used in many domains of non-task-oriented dialogue. For example, the predition and employment of emotions of speakers [34, 104] could be reasonable approaches where they assume emotions can work as relatively universal dialogue states but not given explicitly. Our approach can also be categorized into these approaches that try to design dialogue states for open-domain non-task-oriented dialogue. This study pioneered the use of situations and confirmed its effect in end-to-end dialogue response generation.

As for the persona of speakers, one of situations we targeted, a similar approach that exploits the profile of speakers was investigated [128].

4.3 Conversational situations

Various types of conversational situations could affect our response (or initial utterance) to the listener. Since neural conversational models need massive data to train a reliable model, our study investigates conversational situations that are naturally given or can be identified in an unsupervised manner to make the experimental settings feasible.

In this study, we represent conversational situations as discrete variables. That allows models to handle unseen situations in testing by classifying them into appropriate situation types via distributed representations or the like as described below, and helps to analyze the outputs. We consider the following conversational situations to each utterance and response in our dialogue dataset (Section 4.5), and cluster the situations to assign specific situation types to the utterances and responses in the training data of our conversational models.

Utterance The input utterance (to be responded to by the system) is a primary conversational situation and is already modeled by the encoder in the neural Seq2Seq model. However, we may be able to induce a different aspect of situations that are represented in the utterance but are not captured by the Seq2Seq sequential encoder [95]. We first represent each utterance of utterance-response pairs in our dialogue dataset by a distributed representation obtained by averaging word2vec[1] vectors (pre-trained from our dialogue datasets (Section 4.5.1)) for words in the utterances. The utterances are then classified by *k*-means clustering to identify utterance types.[2]

User (profiles) User characteristics should affect his/her responses as Li et al. [54] have already discussed. We classify profiles provided by each user in our dialogue dataset (Section 4.5.1) to acquire conversational situations specific to the

[1] https://code.google.com/p/word2vec/

[2] We set *k* to 10. Although we also evaluated our method with another *k*, the difference of values does not affect our conclusion.

addressers and listeners. The same as with the input utterance, we first construct a distributed representation of each user's profile by averaging the pre-trained word2vec vectors for verbs, nouns and adjectives in the user profiles. The users are then classified by *k*-means clustering to identify user types.[3]

Time (season) Our utterances can be affected by when we speak as illustrated in Section 4.1, we thus adopted time as one conversational situation. On the basis of timestamp of the utterance and the response in our dataset, we split the conversation data into four season types: namely, spring (Mar. – May.), summer (Jun. – Aug), autumn (Sep. – Nov.), and winter (Dec. – Feb.). This splitting reflects the climate in Japan since our data are in Japanese whose speakers mostly live in Japan.

In training our neural conversational models, we use each of the above conversational situation types for the speaker side and listener (those who respond) side, respectively. Note that the utterance situation is only considered for the speaker side since its response is unseen in response generation. In testing, the conversational situation types for input utterances (or speaker and listener's profiles) are identified by finding the closest centroid obtained by the *k*-means clustering of the utterances (profiles) in the training data.

4.4 Method

Our neural conversational models are based on the Seq2Seq model [106] and integrate mechanisms to incorporate various conversational situations (Section 4.3) at speaker side and listener side. In the following, we briefly introduce the Seq2Seq conversational model [114] and then describe two mechanisms for incorporating conversational situations.

4.4.1 Seq2Seq conversational model

The Seq2Seq conversational model [114] consists of two recurrent neural networks (rnns) called an encoder and a decoder. The encoder takes each word of an

[3] We set *k* to 10, and add another cluster for users whose profiles were not available (6.3% of the users in our datasets).

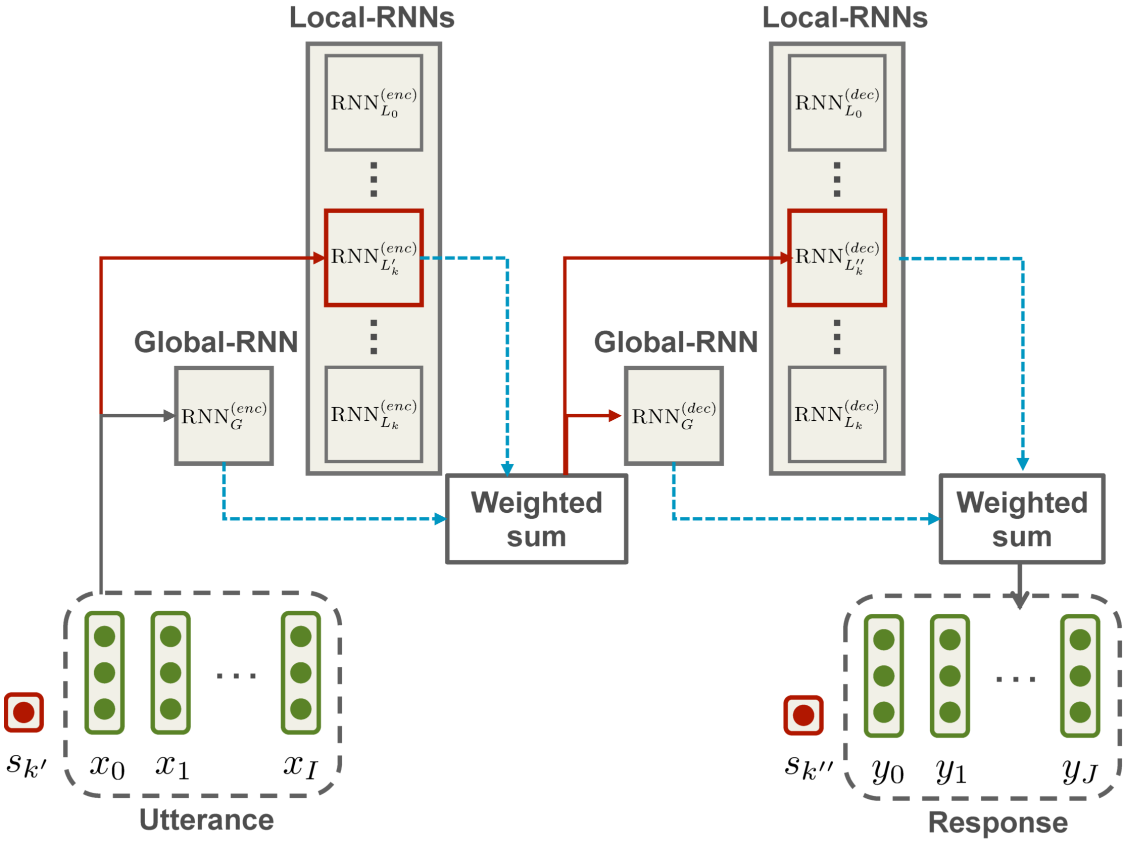
Local-RNNs
$\mathrm{RNN}_{L_0}^{(enc)}$
$\mathrm{RNN}_{L'_k}^{(enc)}$
$\mathrm{RNN}_{L_k}^{(enc)}$
Global-RNN
$\mathrm{RNN}_{G}^{(enc)}$
Weighted sum
$s_{k'}$
x_0
x_1
x_I
Utterance
Local-RNNs
$\mathrm{RNN}_{L_0}^{(dec)}$
$\mathrm{RNN}_{L''_k}^{(dec)}$
$\mathrm{RNN}_{L_k}^{(dec)}$
Global-RNN
$\mathrm{RNN}_{G}^{(dec)}$
Weighted sum
$s_{k''}$
y_0
y_1
y_J
Response

Figure 4.2: Local-global SEQ2SEQ.

utterance as input and encodes the input sequence to a real-valued vector representing the utterance. The decoder then takes the encoded vector as its initial state and continues to generate the most probable next word and to input the word to itself until it finally outputs EOS.

4.4.2 Situation-aware conversational models

The challenge in designing situation-aware neural conversational models is how to inject given conversational situations into RNN encoders or decoders. In this thesis, we present two situation-aware neural conversational models that differ in how strong of an effect a given situation has.

Local-global Seq2Seq

Motivated by a recent success in multi-task learning for a deep neural network [59, 58, 33, 60], our local-global Seq2Seq trains two types of RNN encoder and decoder for modeling situation-specific dialogues and universal dialogues jointly (Figure 4.2).

Local-RNNs are meant to model dialogues in individual conversational situations at both the speaker and listener sides. Each local-RNN is trained (i.e., its parameters are updated) only on dialogues under the corresponding situation. A salient disadvantage of this modeling is that the size of training data given to each local-RNN decreases as the number of situation types increases.

To address this problem, we combine another global-RNN encoder and decoder trained on all the dialogue data and take the weighted sum of the hidden states hs of the two RNNs for both the encoder and decoder to obtain the output as:

$$\begin{aligned} h_i^{(enc)} =& \boldsymbol{W}_G^{(enc)} \textsc{rnn}_G^{(enc)}(h_{i-1}^{(enc)}, x_i) + \\ & \boldsymbol{W}_L^{(enc)} \textsc{rnn}_{L_{k'}}^{(enc)}(h_{i-1}^{(enc)}, x_i), \end{aligned} \tag{4.1}$$

$$\begin{aligned} h_j^{(dec)} =& \boldsymbol{W}_G^{(dec)} \textsc{rnn}_G^{(dec)}(h_{j-1}^{(dec)}, y_{j-1}) + \\ & \boldsymbol{W}_L^{(dec)} \textsc{rnn}_{L_{k''}}^{(dec)}(h_{j-1}^{(dec)}, y_{j-1}), \end{aligned} \tag{4.2}$$

where $\textsc{rnn}_G^{(\cdot)}(\cdot)$ and $\textsc{rnn}_L^{(\cdot)}(\cdot)$ denote global-RNN and local-RNN, respectively, and the $\boldsymbol{W}$s are trainable matrices for the weighted sum. The embedding and softmax layers of the RNNs are shared.

Seq2Seq with situation embeddings

The local-global Seq2Seq assumes that dialogues with different situations involve different domains (or tasks) that are independent of each other. However, this assumption could be too strong in some cases and thus we devise another weakly situation-aware conversational model.

We represent the given situations at speaker and listener sides, $s_{k'}$ and $s_{k''}$, as situation embeddings and then feed them to the encoder and decoder prior to

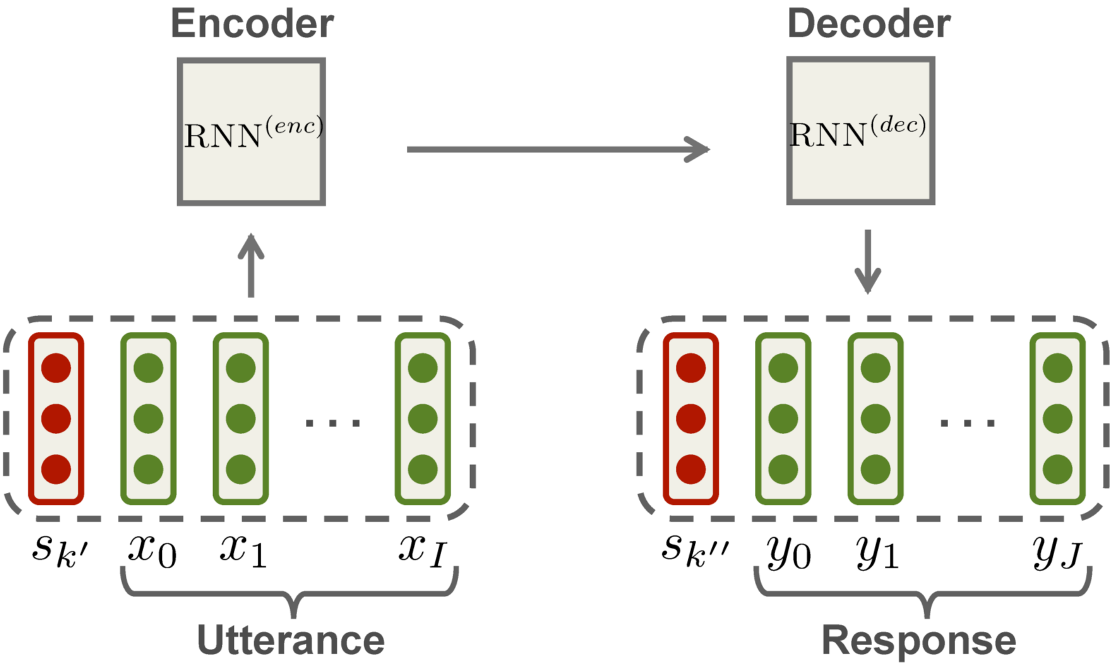

Figure 4.3: Seq2Seq with situation embeddings.

processing sequences (Figure 4.3) as:

$$h_0^{(enc)} = \mathrm{RNN}(h_{init}, s_{k'}), \tag{4.3}$$

$$h_i^{(enc)} = \mathrm{RNN}(h_{i-1}^{(enc)}, x_{i-1}), \tag{4.4}$$

$$h_0^{(dec)} = \mathrm{RNN}(h_{I+1}^{(enc)}, s_{k''}), \tag{4.5}$$

$$h_j^{(dec)} = \mathrm{RNN}(h_{j-1}^{(dec)}, y_{j-1}), \tag{4.6}$$

where h_{init} is a vector filled with zeros and $h_{I+1}^{(enc)}$ is the last hidden state of the encoder.

This encoding was inspired by a neural machine translation system [40] that enables multilingual translation with a single model. Whereas it inputs the target language embedding only to the encoder to control the target language, we input the speaker-side situation to the encoder and the listener-side one to the decoder.

Average length in words (utterances)	15.7
Average length in words (responses)	10.1
Average length in words (user profiles)	37.4
Number of users	386,078

Table 4.1: Statistics of our dialogue datasets (training, validation, and test portions are merged here).

4.5 Evaluation

In this section, we evaluate our situation-aware neural conversational models on massive dialogue data obtained from Twitter. We compare our models (Section 4.4.2) with SEQ2SEQ baseline (Section 4.4.1) using a response selection test instead of evaluating generated responses, since Liu et al. [57] pointed out several problems of existing metrics such as BLEU [84] for evaluating generated responses.

4.5.1 Settings

Data We built massive dialogue datasets from our Twitter archive that have been compiled since March, 2011. In this archive, timelines of about 1.5 million users[4] have been continuously collected with the official API. It is therefore suitable for extracting users' conversations in timelines.

On Twitter, a post (tweet) and a mention to it can be considered as an utterance-response pair. We randomly extracted 23,563,865 and 1,200,000 pairs from dialogues in 2014 as training and validation datasets, and extracted 6000 pairs in 2015 as a test dataset in accordance with the following procedure. Because we want to exclude utterances that need contexts in past dialogue exchanges to respond from our evaluation dataset, we restrict ourselves to only tweets that are not mentions to other tweets (in other words, utterances without past dialogue exchanges are chosen for evaluation). For each utterance-response pair in the test dataset, we randomly chose four (in total, 24,000) responses in 2015 as false response candidates which together constitute five response candidates for the response selection test. Each utterance and response (candidate) is tokenized

[4] Our collection started from 26 popular Japanese users in March 2011, and the user set has iteratively expanded to those who are mentioned or retweeted by already targeted users.

Vocabulary size	100,000
Dropout rate	0.25
Mini-batch size	800
Dimension of embedding vectors	100
Dimension of hidden states	100
Learning rate	1e-4
Number of samples in sampled softmax	512

Table 4.2: Hyperparameters for training.

by MeCab[5] with NEologd[6] dictionary to feed the sequence to the word-based encoder decoder.[7] Table 4.1 shows statistics on our dialogue datasets.

Models In our experiments, we compare our situation-aware neural conversational models (we refer to the former model in Section 4.4.2 as **L/G Seq2Seq** and the latter model in Section 4.4.2 as **Seq2Seq emb**) with situation-unaware **baseline** (Section 4.4.1) for taking each type of conversational situations (Section 4.3) into consideration. We also evaluate **L/G Seq2Seq** without global-rnns (referred to as **L Seq2Seq**) to observe the impact of global-rnns.

We used a long-short term memory (lstm) [126] as the rnn encoder and decoder, sampled softmax [39] to accelerate the training, and TensorFlow[8] to implement the models. Our lstms have three layers and are optimized by Adam [44]. The hyperparameters are fixed as in Table 4.2.

Evaluation procedure We use the above models to rank response candidates for a given utterance in the test set. We compute the averaged cross-entropy loss for words in each response candidate (namely, its perplexity) by giving the candidate following the input utterance to each conversational model, and used the resulting values for ranking candidates to choose top-*k* plausible ones. We adopt **1 in t P@k** [120] as the evaluation metric, which indicates the ratio of

[5] http://taku910.github.io/mecab/

[6] https://github.com/neologd/mecab-ipadic-neologd

[7] The number of words in the utterances and the response candidates in the test set is limited to equal or less than 20, since very long posts do not constitute usual conversation.

[8] https://www.tensorflow.org/

Model	1 in 2 P@1	1 in 5 P@1	1 in 5 P@2
Baseline	64.5%	33.9%	56.6%
Situation: utterance			
L Seq2Seq	67.2%	37.2%	60.6%
L/G Seq2Seq	**68.5%**	**38.2%**	**62.1%**
Seq2Seq emb	65.6%	35.4%	58.2%
Situation: speaker/addressee (profiles)			
L Seq2Seq	67.3%	**38.0%**	60.9%
L/G Seq2Seq	66.4%	36.4%	59.2%
Seq2Seq emb	**67.8%**	37.5%	**61.1%**
Situation: time (season)			
L Seq2Seq	*62.0%*	*30.8%*	*54.8%*
L/G Seq2Seq	65.9%	35.8%	58.1%
Seq2Seq emb	**67.3%**	**37.6%**	**60.7%**

Table 4.3: Results of the response selection test.

utterances that are provided the single ground truth in top k responses chosen from t candidates. Here we use **1 in 2 P@1**,[9] **1 in 5 P@1**, and **1 in 5 P@2**.

4.5.2 Results

Table 4.3 lists the results of the response selection test. The proposed conversational models successfully improved the relevance of selected responses by incorporating conversational situations.

The proposed model that performed best is different depending on the situation type. We found from the dataset that many of the conversations did not seem to be affected by the seasons, that is, time (season) situation is less influential than other situations. This explains the poor performance of **L Seq2Seq** with time (season) situations due to the data sparseness in training local-RNNs, although the sparseness is mostly addressed by global RNNs in **L/G Seq2Seq**.

As stated in Section 4.4.2, **L/G Seq2Seq** is expected to capture situations more strongly than **Seq2Seq emb**. To confirm this, we plotted scattergrams of the utterance vectors (Figure 4.4) and the user profile vectors (Figure 4.5) in the training data by using t-SNE [66]. We provide cluster descriptions by manually

[9] We randomly selected one false response candidate from the four pre-selected ones when $t = 2$.

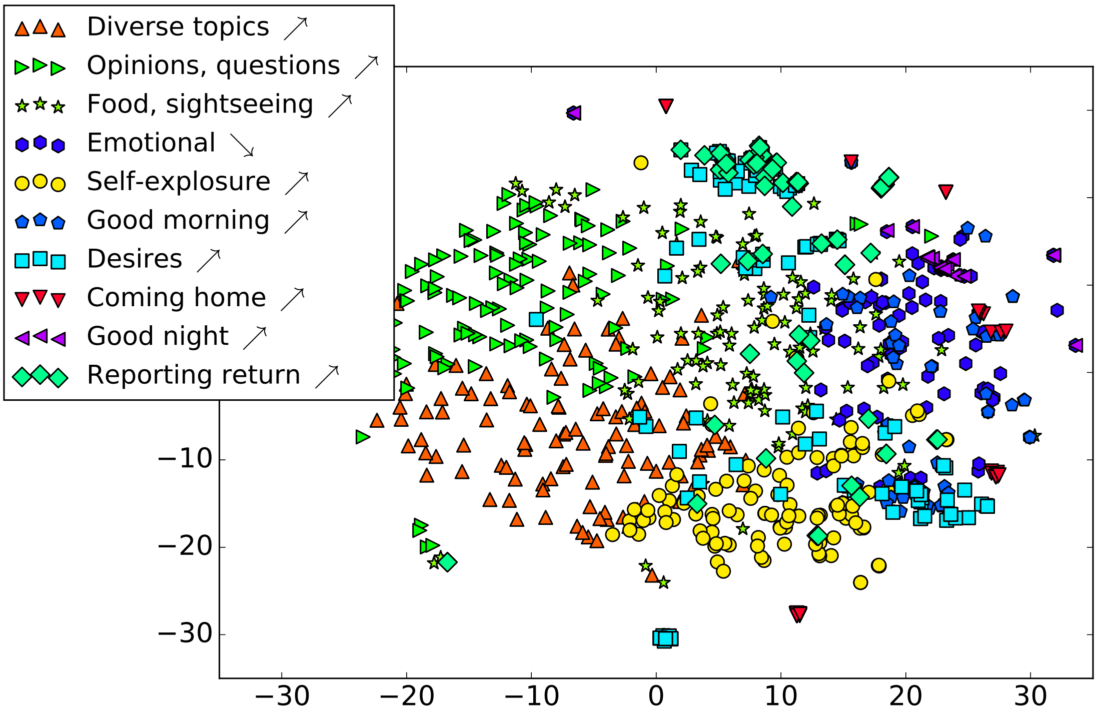

Figure 4.4: The scattergram of sampled utterance vectors visualized using t-SNE.

looking into the content of the utterances and user profiles in each cluster. The descriptions are followed by ↗ if **L/G Seq2Seq** performed better than **Seq2Seq emb** in terms of 1 in 5 P@1 for test utterances with the corresponding situation type, by ↘ if the opposite and by → if comparable (differences are within ± 1.0%). Elements of clusters were randomly sampled.

L/G Seq2Seq tends to perform better for utterances with densely concentrated (or coherent) speaker profile clusters (Figure 4.5). This is because utterances given by the speakers in these coherent clusters (and the associated responses) have similar conversations, situations of which are captured by local-rnns in the local-global Seq2Seq.

This explains the reason why **L/G Seq2Seq** outperformed the other situation-aware conversational models when utterance situations are considered (Figure 4.4). Conversations in the same clusters are naturally consistent, and conversations assigned to the same clusters form typical activities or specific tasks (*e.g.*, greetings, following other users, and questions (and answering)) in Twitter conversation.

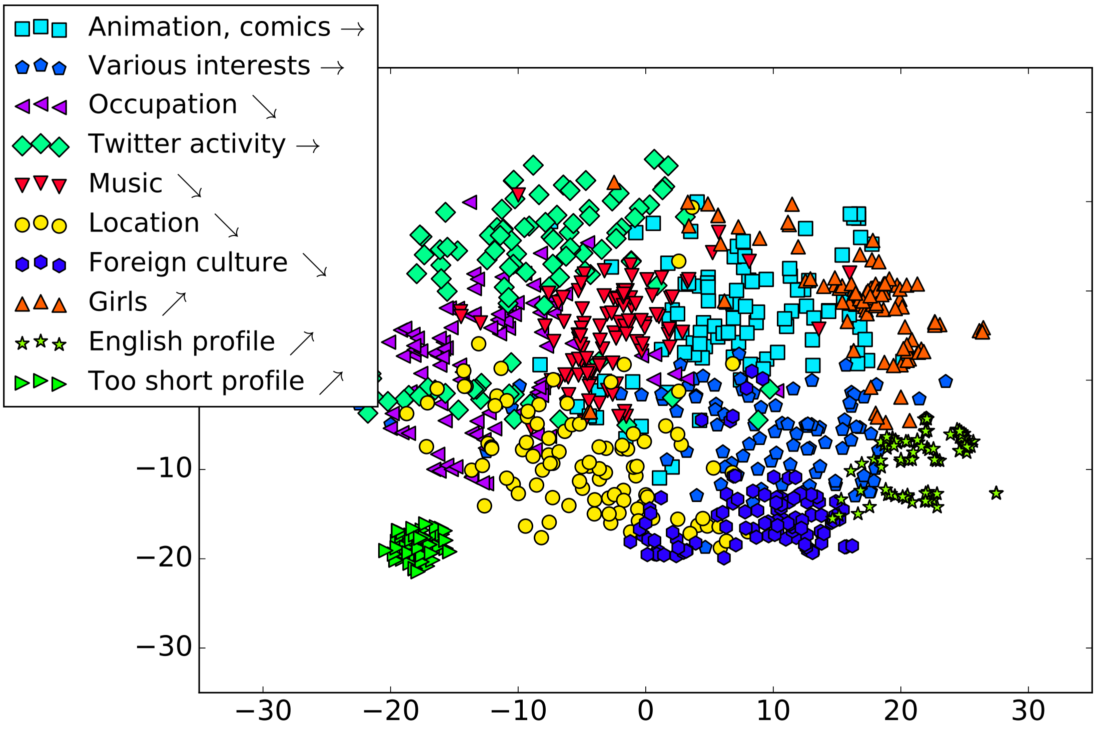

Figure 4.5: The scattergram of sampled user profile vectors visualized using t-SNE.

L/G Seq2Seq, designed as a kind of multi-task Seq2Seq, literally captures these task-specific behaviors in the conversations.

Although some utterance clusters have general conversations (*e.g.*, diverse topics), the response performances in those clusters have still improved. This is because these general clusters are free from harmful common responses that are quarantined into situation-specific clusters (*e.g.*, greetings etc.) and the corresponding local-RNNs should avoid generating those common responses. Note that this problem has been pointed out and addressed by Li et al. [53] in a totally different way.

Examples Table 4.4 lists the response candidate selected by the baseline and our models. As we had expected, the situation-aware conversational models are better at selecting ground-truth responses for situation-specific conversations.

Situation:	utterance (opinions, questions)
Input:	ちょっと最近BOTのフォロー多いんですけど (I've recently been followed by many bot accounts.)
Baseline	お疲れ様やで(You've gotta be tired.)
L/G Seq2Seq	ブロックしちゃいましょう(Let's block them.)
Situation:	addressee profiles (girls)
Input:	なにグラブル始めてるんだ原稿しろ (Why am I starting Granblue Fantasy? I have to write the paper...)
Baseline	おい、大丈夫か？(Hey, are you okay?)
Seq2Seq emb	フレンドなろ♡ (Let's be friends♡)
Situation:	time (season) (summer)
Input:	7月になって、流石にパーカーは暑くなってきた (July is too warm to wear a hoodie.)
Baseline	そうなんです! (Yes!)
Seq2Seq emb	まだ着てたの!? (Do you still wear one?)

Table 4.4: Responses selected by the systems.

4.6 Chapter Summary

We proposed two situation-aware neural conversational models that have general mechanisms for handling various conversational situations represented by discrete variables (Section 4.4.2): (1) local-global Seq2Seq that combines two Seq2Seq models to handle situation-specific dialogues and universal dialogues jointly, and (2) Seq2Seq with situation embeddings that feeds the situations directly to a Seq2Seq model. The response selection tests on massive Twitter datasets confirmed the effectiveness of using situations such as utterances, user (profiles), or time.

Although we evaluated our method in dialogue response generation where situations were explicitly available and likely to affect responses, the problem and solution discussed in this study are applicable in other text generation task. Even in machine translation, the task where the diversity of outputs is relatively low, personalized machine translation have been explored [72, 87, 122, 69]. We consider their focuses and ideas were partially similar to ours although the types of situations are limited due to the difficulty in obtaining situations in such tasks.

Chapter 5

Speculative Latent Variables Sampling for Handling Latent Domains

5.1 Introduction

In early neural-based approaches in dialogue modeling, it has been a serious problem that employing a simple encoder-decoder framework tended to generate safe responses (a.k.a, dull responses) [53]. The first reason is that dialogue datasets often contain short, trivial, and frequent utterances and responses such as greetings. In training with such a dataset, it is an easy way for a model to simply imitate such frequent responses for optimizing a training objective. The second and more important reason is the nature of conversations. In dialogue, there often exists many appropriate responses to a given untterance. However, conventional encoder-decoder frameworks have modeled response generation as a one-to-one projection from an input sentence to an output sentence [15, 106, 105]. The contradiction appears as the safe response problem; there is no reason for a model to choose a specific and interesting response from probable responses. As a result, it simply imitates a frequent and versatile response in training data.

The use of variational model is a promising approach to resolve the problem. This was inspired by the success of Variational Auto-Encoder (VAE) in the field of computer vision [46]. Variational dialogue models estimate the prior and posterior distributions of a conversation and latent variables sampled from the distributions perform as randomness. The latent variables can work as additional clues to generate a response with high context-/domain-/style- specificity.

While various variational attempts have been made for diversifying the output of dialogue models [9, 45, 100, 129], it is still unclear what the latent space of a model captures from the randomness in conversations. One reason is that sampled latent variables and the parameters of the distributions are continuous vectors approximated by an neural network, which are difficult to analyze. Moreover, in traditional variational models, latent variables corresponding to the same response can be scattered throughout the model's prior distributions and are less structured.

In this study, we aim to organize the latent space by a simple method focusing on the sampling of latent variables in training. We consider the main reason of the disorganization in latent spaces is the discrepancy between a sampled latent variable and a training example. To resolve that, our proposed model speculatively samples multiple latent variables and exploits only the one for optimization that leads to the minimum loss.

5.2 Related work

Major existing approaches to preventing the generation of safe responses can be broadly categorized into three types: 1) using additional contexts, 2) changing decoding objectives, and 3) employing variational models. Here, we review the overview of the approaches.

The first approch is represented as personalization. It is based on the idea that by giving more information to the model and by making the context more specific, the response will also obtain high specificity. To controll responses to be generated, they relied on the speaker's persona [128, 68, 18, 6], emotions to be expressed [34, 130], topics [124], and situations [96] of a conversation as additional contexts.

In the second approach, the criterion for response generation is explicitly controlled by changing the decoding method. Concretely, they employed mutual information [53], inverse token frequency [73], topic or semantic similarity between input and output [5].

Our method belongs to the third approach. We here review existing methods for variational response generation. With the success of variational recurrent hierarchical encoder-decoder (VHRED) in dialogue modeling [101], variational models have been expected as one of the solutions to the safe response problem. However, the difficulty in optimization of variational models raised another problem – KL vanishing (or latent variable vanishing). With a sufficient number of parameters, the use of a powerful network allows a model to pay attention to the representations of the encoder and decoder. As the result, the model can totally ignore latent variables and the prior and posterior distribution become almost the same. Thus, many existing studies have proposed a method to control the optimization of KL-divergence (Kullback–Leibler divergence) [9, 129, 45, 103, 55, 31, 25, 27, 28]. Many of these approaches were based on regularization.

To design a latent space where the relevance and diversity of outputs are geometrically reflected, Gao et al. [27] proposed SPACEFUSION, where regularization terms are added to training objectives that fuse and interpolate the latent space. Among the aforementioned existing studies, their approach shares with us a similar goal of organizing the latent space by the meaning of output. While their method was based on the regularization to latent variables and requires a modification to model's architecture, we propose a model-agnostic method that focuses only on the sampling of latent variables and optimization. Kruengkrai [50] proposed to sample multiple latent variables in text modeling, similarly to our method. However, the intention is different as we aim to control sampling procedures based on training losses to provide suitable latent variables. We will also evaluate their method and discuss the difference in Section 5.4.

5.3 Speculative Latent Variables Sampling

5.3.1 Preliminary: T-CVAE

Before discussing our proposed method, we first introduce Transformer-based conditioned variable auto-encoder (T-CVAE) that we employed as a core architecture of our model [116]. Namely, T-CVAE was one of conditioned variational

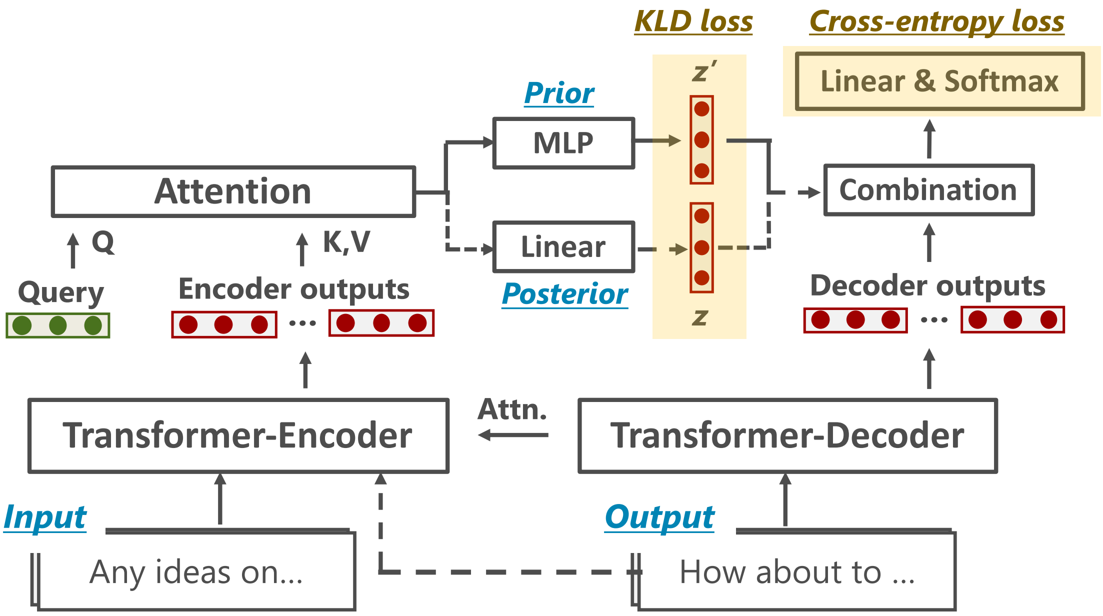

Figure 5.1: The overview of T-CVAE. Dotted lines represent computations only in training.

auto-encoders (CVAE) proposed for story complilation. The architecture is based on Transformer [112], a recent promising architecture. The main purpose is to handle uncertainty in text generation, similarly to Latent Variable Hierarchical Recurrent Encoder-Decoder (VHRED) [101] .

The optimization of T-CVAE is similar to CVAE; we maxmize the evidence lower bound (ELBO) instead of directly optimizating the conditional probability $\log p(y \mid x)$ as follows:

$$\begin{aligned}\log p(y \mid x) &= \log \int_z p(y \mid x,z)p(z \mid x)dz \\ &\geq \mathbb{E}_{q(z|x,y)}[\log p(y \mid x,z)] \\ &\quad - D_{KL}(q(z \mid x,y) || p(z \mid x))\end{aligned} \tag{5.1}$$

Here, we denote an inputm an output, and a latent variable by x, y, and z, respectively. The prior and posterior distributions are denoted by $p(z \mid x)$ and $q(z \mid x,y)$, respectively. In training, it computes the posterior distribution from both a given input x and a given output y. And then, it samples a latent variable z

and compute cross-entropy loss from $q(y \mid x,z)$. In addition, it computes the prior distribution $p(z \mid x)$ only from the given input x and adds the KL-divergence D_{KL} to the loss.

Practically, the prior and the posterior distribution are supposed to be a multivariate normal distribution whose covariance matrix is constrained to be a diagonal matrix for convenience of computation.

The mean μ and covariance $\log\left(\sigma^2\right)$ of the prior distribution $p(z \mid x) \sim N\left(\mu', \sigma'^2 \mathrm{I}\right)$ are computed as:

$$\begin{aligned} h &= \mathrm{Attention}\left(c, E^L_{out}(x), E^L_{out}(x)\right) \\ \begin{bmatrix} \mu \\ \log\left(\sigma^2\right) \end{bmatrix} &= \mathrm{MLP}_p\left(h\right) \end{aligned} \tag{5.2}$$

Similarly, the mean μ' and covariance $\log\left(\sigma'^2\right)$ of the posterior distribution $q(z \mid x,y) \sim N\left(\mu, \sigma^2 \mathrm{I}\right)$ are computed as:

$$\begin{aligned} h' &= \mathrm{Attention}\left(c, E^L_{\mathrm{out}}(x;y), E^L_{\mathrm{out}}(x;y)\right) \\ \begin{bmatrix} \mu' \\ \log\left(\sigma'^2\right) \end{bmatrix} &= h' W_q + b_q \end{aligned} \tag{5.3}$$

The differences in the two equations above are: 1. networks to induce the distributions and 2. whether or not the output y is used for computation. The outputs from the last encoder is denoted as $E^L_{\mathrm{out}}(x;y)$ and $x;y$ means concatenation of the input and output. Attention(Q,K,V) means applying attention mechanism to summarize the outputs from encoder. Q, K, and V are the query/key/value vector, respectively. In the original T-CVAE [116], MLP_p consists of two feed-forward layers.

$\mathrm{MLP}_*(\cdot)$ denotes a multi-layered perceptron.

After estimating the prior and the posterior distributions, T-CVAE conducts a decoding process with a sampled latent variable z, similarly to CVAE. The computation in decoder is defined as:

$$\begin{aligned}
C_t &= \tanh\left(\left[z; D^L_{\text{out},t}\right] W_c\right) \\
O_t &= \text{MLP}_o\left(C_t\right) \\
P_t &= \text{softmax}\left(O_t\right)
\end{aligned} \tag{5.4}$$

$D^L_{\text{out},t}$ denotes outputs from the last decoder at time step t. As Eq. (5.4) shows, the sampled latent variable z is simply concatenated with $D^L_{\text{out},t}$ just before the output layer. In other words, computations in the decoder is the same as the original Transformer. Additionally, it should be mentioned that the FFN and multi-head attention parameters are shared between the respective encoder and decoder layers. This is because the architecture was proposed for story compilation. Differently from machine translation, it was assumed that the language was the same across an input and an output. We also adopt this setting in the following experiments since sharing encoder and decoder can reduce the number of trainable parameters.

5.3.2 Speculative Latent Variables Sampling

The problem we address in this work is the independence between the sampling of latent variable and a given example during training. As explained in Section 5.3, in the training of a typical variational model, the model computes prior and posterior distributions $p(z \mid x)$ and $q(z \mid x, y)$ from the utterance x and response y in a conversation given as an example. And then, the training objective is optimized; the objective is typically the sum of the cross-entropy loss and the KL-divergence between the prior and posterior distributions.

Ideally, the prior distribution should cover all possible responses to a given context x while the response y is fixed when the posterior distribution is induced. Here, what if a latent variable sampled from the posterior distribution is inappropriate to represent the observed response? For example, suppose that there is a utterance-response pair, “Any musicians to recommend?” and “Why don’t you listen to MJ?” (Figure 5.2). Although the posterior distribution is induced under the observation of the given response, it can produce a latent variable that leads to another probable response, “Jimi Hendrix is a rock guitar legend.” The reason

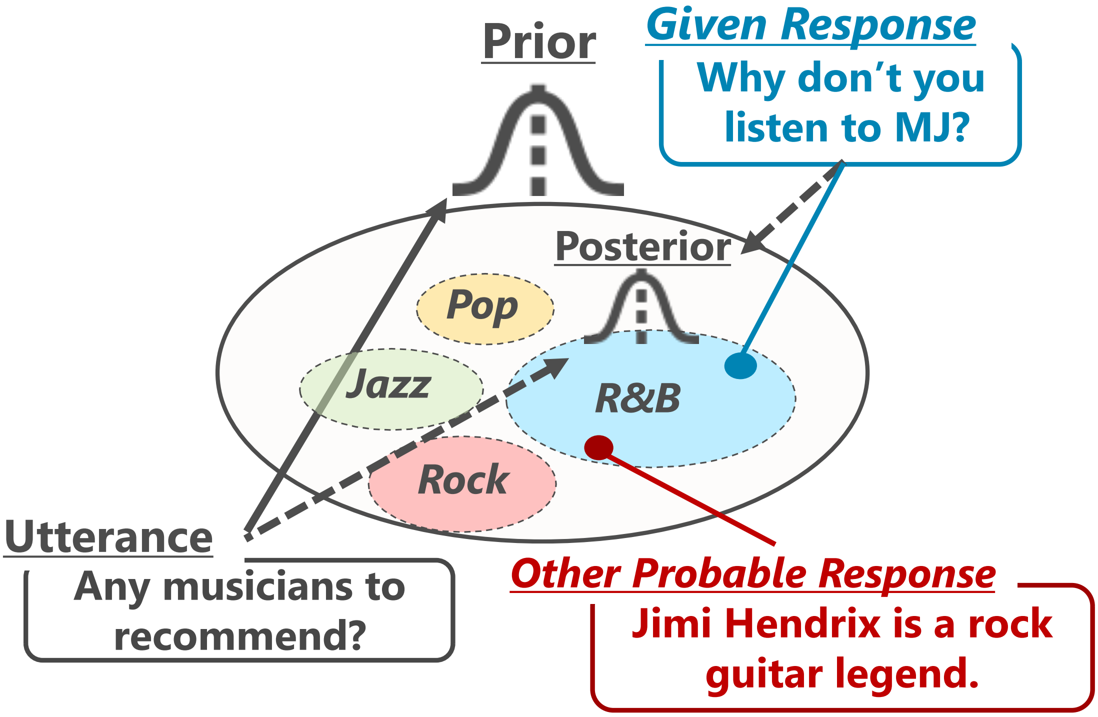
Prior
Given Response
Why don't you listen to MJ?
Posterior
Pop
Jazz
R&B
Rock
Utterance
Any musicians to recommend?
Other Probable Response
Jimi Hendrix is a rock guitar legend.

Figure 5.2: The overview of the problem in the training of CVAE-based models: the posterior can produce a variable leading to another probable response.

is for the confusion is that parameters of a model are incomplete during training, and thus the estimation of the posterior distribution can fail. Moreover, due to the optimization of KL-divergence, the posterior distribution is encouraged to be similar to the prior distribution that covers other probable responses too.

In this situation, the optimization becomes skewed; while the model tries to minimize the cross-entropy loss computed from a given input, a given output, and a sampled latent variable, the sampled latent variable might not represent the given output in the latent space of the model. As the result of the discrepancy, the model can lose track of the correspondence between a sampled latent variable and the response to be generated, and the latent space becomes disorganized.

To resolve the problem, we propose a quite simple method to disentangle the discrepancy. Part of our method was inspired by dynamic oracle [29] that allows a shift-reduce dependency parser to choose an easy-to-decode oracle operation among all the possible oracle operations that will ultimately reach the

# encoder/decoder layers	6	Label smoothing rate	0.1
# attention heads	8	Init. learning rate	1e-3
Dim. of embeddings	512	(warmup)	1e-7
Dim. of Transformer	2048	Dropout rate	0.1
Vocab. size (enc&dec)	16k	Beam size for decoding	5
Max. tokens in batch	40k	Length penalty	1.2

Table 5.1: Hyperparameters of models.

gold dependency structure. Analogously, we aim to provide the most probable latent variables that can reach a given response in training.

Specifically, we sample multiple latent variables $\{z_0, z_1, \cdots, z_{N-1}\}$ from the posterior distribution and compute the loss respectively in training. And then, we compute the gradients from only the sampled latent variable that leads to the smallest loss. We minimize the training objective $L(\hat{z})$ defined as:

$$\begin{aligned} L(z) &= -\mathbb{E}_{q(z|x,y)}[\log p(y \mid x, z)] + D_{KL}\left(q(z \mid x, y) \| p(z \mid x)\right) \\ \hat{z} &= \operatorname*{argmin}_{z_i} L(z_i) \end{aligned} \tag{5.5}$$

The objective is the same as Eq. (5.1) while the selection of $\hat{z}$ is only the difference. Note that our method affects only the training of models. In testing, the decoding process is identical to conventional variational models; they sample one or several latent variables, computes the probability distribution, and chooses the next token according to the distribution.

5.4 Evaluation in Dialogue Response Generation

5.4.1 Models

We evaluate the effect of the proposed method in dialogue response generation. We adopted subword-based T-CVAE [116] that we implemented in fairseq (v0.8.0)[1] [83], as the core architecture for the dialogue models. We basically followed the major hyperparameters of Transformer-base [112], as shown in Table 5.1. Following the original settings, we shared the parameters of feed-

[1] https://github.com/pytorch/fairseq

forward layers and attention mechanism between the encoder and decoder. The embedding layers of the encoder and the decoder are also shared. Note that an input utterance and an output response are encoded by the same encoder for inducing the distributions.

We compared the following methods that aim to control the latent space of variational models.

T-CVAE: refer to Section 5.3.1.

T-CVAE + Monotonic KL annealing: the optimization of KL-divergence is linearly annealed for preventing KL vanishing [9]. Concretely, the second term in Eq. (5.1) is weighted by λ that begins from 0 and is linearly increased to 1 for the first epoch.

T-CVAE + Cyclical KL annealing: although the KL-divergence term is annealed similarly to **Monotonic KL annealing**, the annealing scheme is namely cyclical [25]. During the first half of an epoch, we start with $\lambda = 0$ and linearly increase λ to 1. λ stays at 1 during the second half of the epoch.

SPACEFUSION: we add another loss for fusing the vector space of input utterances and output responses, to the cross-entropy loss [27]. The aim of this model to modulate the latent space is related to our approach. Note that the covariance of Gaussian distribution of this model is not parametrized and thus only this model is slightly different from **T-CVAE**.

T-CVAE + BoW loss: we add the bag-of-word (BoW) loss to the training objective in Eq. (5.1) [129]. This is a loss to prevent a model from ignoring the latent variable by a constraint that ties up a sampled latent variable with a given response. Namely, the latent variable is also used for predicting bag-of-words in the response as another task, which makes the latent variable capture global information about the response.

T-CVAE + Monte Carlo sampling: similarly to the speculative sampling, we sample multiple latent variables and use the average for computing the loss in training [50].[2] Note that the original method, LSTM-VAE-AVG, takes the average of the decoder's outputs computed from each sampled latent variable. In T-CVAE, however, latent variables do not affect the computation in the Transformer-decoder stacks and they are concatenated with the decoder's output before the output layer instead. Thus, taking the average of the latent variables in T-CVAE is equivalent to the procedure in the original model.

T-CVAE + Speculative sampling: refer to Section 5.3.2. We sample five latent variables for each conversation considering computational costs.[3]

5.4.2 Dataset and Preprocessing

Similarly to Section 4.5.1, we constructed a Japanese dialogue dataset from our Twitter archive while simulating a tweet and subsequent mentions between two users as a conversation. Hereafter, we refer to messages sent to other users as *mentions*, and otherwise as *tweets*. Other types of messages such as retweets (RT) or quote tweets (QT) were excluded from collection in advance. The concrete process for building the dataset is as follows.

First, we collected messages from the Twitter archive in 2017 and 2018 for training and development. We used the messages in 2019 for testing. In the preprocessing to each message, we removed emoticons, hashtags (*#hashtag*) at the end of a message, and usernames (*@username*) of Twitter at the beginning of a mention from all messages. To reduce noises in conversations, we filtered out messages that satisfied one of the following conditions. Note that part of the message-level preprocessing and filtering was based on Adiwardana et al. [1].

- If a message contained URLs or several words implying it had additional contexts outside the message such as "RT," "QT," etc.
- If a message was made by users whose username or profile contained "bot."
- If a message contained a hashtag or a username in the middle of it because removing it can make the message ungrammatical.

[2] We set the number of sampling to five as same as the speculative sampling.

[3] We also trained models where the number of sampling is 2, 5, 10, or 20. The tendency of the results was similar and we select five as the number of sampling for the main results from the validation results.

Utterance	**Response**
Accepted conversations	
インターンシップで服装自由って言われた時どうすりゃええねん	フルスーツ貸しましょうか?
やってしまった. . . 大きく足首をひねってしまった... 歩くのがきつい... だが病院はあいていない...	そーゆー時は救急車呼んで全然オッケーですから...!!
今更PS4を買おうかと思ってるのですが、500GBと1TBどっちがいいの?? 誰か教えて。	HDがいっぱいになる前に本体壊れるので少なくていいんじゃないかなーって思います笑
Rejected conversations	
おはよー	おはよん
当選おめでとうは!!!?!?!?	おめでとうございます
フックでうらどうするの? クソ強ドローぶち当て? 気合い使ったりなんかな?	ドローで牽制牽制。あんまりストレートは振っちゃ駄目 気合いはWR専用とエンゲツトウで無かったことにされるんで歌えですかね

Table 5.2: Examples of accepted/rejected conversations in filtering for human evaluation. Rejected conversations were too short, generic, or difficult for annotators to imagine the context behind them.

- If a preprocessed message was identical to another message in the same month since such a message could often be generated by a bot or could be trivial like greetings.
- If the the percentage of the same character in a message was more than 50%.

And then, we collected conversations starting from a tweet between two users. The reason why we excluded conversations starting from a mention was that they could include implicit contexts that had been made before the conversation. The numbers of examples were 18,116,756 for training, 191,890 for development, and 96,276 for testing.

Moreover, we sampled 100 conversations from the testing dataset for human evaluation by manual filtering with observing only the contexts and the reference responses. This was because human evaluation is costly and the randomly selected testing data includes 1) immoral conversations, 2) trivial conversations

	BLEU	dist-1	dist-2	Average Length
Reference	-	4.11	30.60	12.01
T-CVAE	2.48	1.14	4.24	15.68
T-CVAE + Monotonic KL annealing	2.73	1.27	4.47	13.99
T-CVAE + Cyclical KL annealing	2.73	1.19	4.24	14.01
SPACEFUSION	2.86	1.50	4.74	8.27
T-CVAE + BoW loss	1.97	1.56	8.74	10.18
T-CVAE + Monte Carlo sampling	0.53	**1.79**	**18.25**	21.20
T-CVAE + Speculative sampling	**2.91**	1.57	6.48	11.31

Table 5.3: Results of automatic evaluation.

such as short greetings, 3) highly specialized conversations, and 4) conversations that have many undescribed contexts outside the text (e.g., subsequent comments to his/her retweet not as a mention, conversations about an image a speaker uploaded, etc.). Such conversations are too difficult even for human evaluators to judge what can be a good response, which hinder us from conducting solid evaluations and analyses. As it is difficult to completely remove them only by automatic preprocessing, we manually chose conversations that were suited to human evaluation. In Table 5.2, we show several examples of accepted/rejected conversations in data filtering for human evaluation.

The constructed dataset was tokenized by MeCab.[4] And then, we train a subword tokenization model with SentencePiece (v0.1.83)[5] from randomly sampled 1,000,000 sentences from the training data. [52] Similarly, we trained CBOW vectors[6] of subwords from another 1,000,000 sentences randomly sampled from training data for initialization of the models' embedding layers. [71]

5.4.3 Automatic Evaluation

For automatic evalaution, we employed several common metrics: case-sensitive BLEU [84], dist-*n* [53], and the average length of outputs. Table 5.3 shows the results.

First, compared to the vanilla **T-CVAE**, the BLEU scores of the models employing the methods for preventing KL-vanishing (hereafter, advanced models)

[4] https://github.com/taku910/mecab
[5] https://github.com/google/sentencepiece
[6] https://code.google.com/archive/p/word2vec/

	BLEU	dist-1	dist-2	Avg. length
T-CVAE + Speculative sampling ($K = 2$**)**	2.69	1.28	4.95	13.61
T-CVAE + Speculative sampling ($K = 5$**)**	2.91	1.57	6.48	11.31
T-CVAE + Speculative sampling ($K = 10$**)**	2.70	1.54	7.00	11.47
T-CVAE + Speculative sampling ($K = 20$**)**	2.40	1.51	7.28	10.64

Table 5.4: Automatic evaluation results of our proposed model with different K.

increased except **T-CVAE + BoW Loss** and **T-CVAE + Monte Carlo sampling**. This is a side effect of the methods; we observed that the vanilla T-CVAE tend to generate noisy or ungrammatical sentences containing repetitions. The repetations sometimes appeared in **T-CVAE**, **T-CVAE + Monotonic KL annealing**, and **T-CVAE + Monotonic KL annealing**. The relatively longer average length of the models compared to **Reference** also shows that. The BLEU scores in the experiments show that responses generated by the advanced models with a higher BLEU score were simply more grammatical rather than human-like. **SPACEFUSION** was relatively grammatical and did not contain the too long repetition. However, the outputs tended to be short and the improvement of diversity was modest. In the proposed models, we consider that variational response generation worked well and improved the skewed training, where models try to learn one-to-one mappings from one-to-many conversational data.

Second, **T-CVAE + BoW loss**, **T-CVAE + Monte Carlo sampling**, and **T-CVAE + proposed** achieved remarkably high dist-1 and dist-2, which represents high diversity of the generated responses. However, the outputs from **T-CVAE + BoW loss** and **T-CVAE + Monte Carlo sampling** were noisy and less relevant to the context as the low BLEU score shows. We will discuss the reason of the tendency with the results in the following sections (Section 5.4.4 and Section 5.4.5).

Although we define $K = 5$ as the hyperparameter of the main proposed model that was manually evaluated, we also show the results of the proposed method with different k in Table 5.4. The model with $K = 2$ was similar to **T-CVAE**, although all metrics were improved. The specificity of the models with relatively larger $K = \{10, 20\}$ was high as shown by the dist-2 scores.

Overall, we conclude that our simple modification, the speculative latent variables sampling worked well and made the generated outputs more diverse without decreasing the BLEU score.

	Sensibleness	Specificity	Average
Reference	4.67	4.33	4.50
T-CVAE	3.58	1.35	2.46
T-CVAE + Monotonic KL annealing	3.49	1.22	2.35
T-CVAE + Cyclical KL annealing	3.58	1.29	2.44
SPACEFUSION	3.66	1.42	2.54
T-CVAE + BoW loss	3.04	**1.58**	2.31
T-CVAE + Monte Carlo sampling	1.42	0.70	1.06
T-CVAE + Speculative sampling	**3.94**	1.52	**2.73**

Table 5.5: Results of human evaluation. Pearson correlation coefficient between evaluators was 0.69.

	KL-divergence	$\|\|\mu\|\|$	$\|\|\sigma\|\|$
T-CVAE	0.003	0.240	21.742
T-CVAE + Monotonic KL annealing	0.060	0.216	15.155
T-CVAE + Cyclical KL annealing	0.088	0.330	15.268
T-CVAE + BoW loss	24.073	36.949	92.003
T-CVAE + Monte Carlo sampling	11.826	1.526	19.814
T-CVAE + Speculative sampling ($K = 2$)	0.621	0.638	21.746
T-CVAE + Speculative sampling ($K = 5$)	1.573	0.953	20.949
T-CVAE + Speculative sampling ($K = 10$)	2.192	1.053	20.347
T-CVAE + Speculative sampling ($K = 20$)	2.910	1.120	18.743

Table 5.6: Average KL-divergence, norm of mean vector μ, and norm of standard deviation vector σ for prior distribution in validation.

5.4.4 Human Evaluation

We also conducted human evaluation shown in Table 5.5 following Adiwardana et al. [1]. In the evaluation, annotators provided scores from 1 to 5 for each anonymized response from the point of view of 1) **sensibleness** and 2) **specificity**. Both metrics literally show how sensible and specific to a given context a response was. In human evaluation, there could be responses that are less relevant to the context and difficult to assess whether they are specific or noisy. Therefore, we allowed annotators to label a response as unevaluable and such a response is scored as specificity=0 in this experiment.

The tendency of the results was similar to that of Section 5.4.3. **SPACEFUSION** and **T-CVAE + Speculative sampling** achieved relatively high sensibleness (i.e., relatedness to the context). The specificity of **T-CVAE + BoW loss** and **T-CVAE**

+ Speculative sampling were remarkably higher than other models while the sensibleness of **T-CVAE + BoW loss** was low. The specificity of **T-CVAE + Monte Carlo sampling** was largely decreased because its outputs were too irrevant and tended to be regarded as unevaluable (specificity=0) by the annotators. We consider the reason for the decrease in sensibleness compared to **T-CVAE** as follows. **T-CVAE + BoW loss** worked too strongly as a constraint on the latent space and the distributions became enlarged. In **T-CVAE + Monte Carlo sampling**, latent variables close to the mean of the posterior distribution were more likely to be trained. As a result, in testing, it is possible for the two models to sample latent variables from unreliable regions that were not optimized enough.

Overall, the proposed model achieved comparable results in both sensibleness and specificity. While the compared methods could lose the sensibleness of generated responses in return for the high specificities, the sensibleness of **T-CVAE + Speculative sampling** was still kept.

5.4.5 Statistics of Distributions

Table 5.6 shows the average KL-divergence and the average norms of the mean and variance of the prior distribution for each model in validation.[7] The KL-divergence of the vanilla **T-CVAE** almost vanished and the employmennt of KL annealing methods slightly mitigated that.

Employing **BoW loss** significantly increased the KL divergence and the norms of μ and σ. This is because **BoW loss** worked as a quite strong restriction in training that required a sampled latent variable to predict all the tokens in the output sentence. As a result, the prior distribution was extremely enlarged and a latent variable sampled from the distribution could lead to generating less sensible and relevant responses as shown in Table 5.5.

Meanwhile, the proposed model successfully prevented KL vanishing without significantly enlarging the distribution. Interestingly, the larger K we chose, the KL-divergence also became large. We consider that the tendency support our hypothesis discussed in Section 5.3.2. Speculatively sampling latent variables

[7] **SPACEFUSION** does not have parametrized distributions. Instead, a Gaussian noise sampled from a static distribution was added to outputs from the encoder.

Input Utterance	北陸にお住まいの、あるいは北陸出身の方に訊きたいのですが2月末の福井って気温的にはどうなんですか厚着してったほうがいいんでしょうか (*I would like to ask someone who lives in Hokuriku or is from Hokuriku. What is the temperature of Fukui at the end of February? Should I wear thick clothes?*)
Reference Response	本州は最近寒いようじゃな (*Honshu would be cold these days.*)
T-CVAE	北海道の方がいいと思いますよ。 (*I guess Hokkaido is better.*)
T-CVAE + Monotonic KL annealing	上着でいいと思いますよ! (*I think you should wear a jacket!*)
T-CVAE + Cyclical KL annealing	北海道の方が暖かいですよ。 (*Hokkaido is warmer.*)
SPACEFUSION	雪が降ってます。(*It's snowing there.*)
T-CVAE + BoW loss	山手線は北上しないと寒いです。 (*The Yamanote line is cold unless you go north.*)
T-CVAE + Monte Carlo sampling	私もつかの間が感じられませんのでご気持ちはわかりますが冬といえばすごく暖かいです (*I can't feel the moment and I understand how you feel, but winter is very warm.*)
T-CVAE + Speculative sampling	普通に寒いと思いますよ。 (*I think it's usually cold there.*)
Input Utterance	急募 喉の痛みの緩和方法 (*Any ideas on how to relieve sore throat?*)
Reference Response	マヌカハニーを舐める (*Eat Manuka honey.*)
T-CVAE	病院に行った方がいいですよ。 (*You should go to a hospital.*)
T-CVAE + Monotonic KL annealing	葛根湯を飲んでみては? (*Why don't you try Kakkonto?*)
T-CVAE + Cyclical KL annealing	お大事にしてください...! (*I hope you get well soon.*)
SPACEFUSION	お大事になさってください... (*I hope you get well soon.*)
T-CVAE + BoW loss	胃腸炎にならなくていいと思います。 (*I think you don't have to have gastroenteritis.*)
T-CVAE + Monte Carlo sampling	自分のやつっ！ (*Your own!*)
T-CVAE + Speculative sampling	ビタミンCを摂るといいよ。(*Take vitamin C.*)

Table 5.7: Examples of generated outputs in dialogue response generation.

and selecting the most probable one for optimization resolved the discrepancy between a given conversation and a sampled latent variable.

5.4.6 Output Examples

Table 5.7 shows several examples of the generated responses. As demonstrated in many existing studies, responses generated by a vanilla T-CVAE still contained a safe response or repetations of a same phrase in the context. Although this tendency was also observed in T-CVAE-based models, the diversity was slightly improved. However, it also made responses less context-specific as the first example shows. This was due to the trade-off between relevance and specificity discussed in existing studies [27].

We often observed that the proposed method helped model avoid the repetition of context words. As represented in the examples, the generated responses by the proposed model tended to contain more topic-specific words such as "ビタミンC (vitamin C)".

5.5 Evaluation in Machine Translation

Dialogue response generation is a task where the diversity of outputs is likely to affect a model's performance. However, also in other generation tasks, many-to-many relations between inputs and outputs can be similarly problematic to a varying degree. The ability of models to generate diverse output candidates is beneficial. For example, in machine translation, where probable outputs are relatively strictly restricted by the inputs, there still can exist many probable outputs depending on the situation of the inputs or the number of patterns with the same meaning in the target language. We also evaluated the proposed method in machine translation to investigate how variational encoder-decoder models perform in such tasks from the viewpoints of quality and diversity.

5.5.1 Models

We employed two NMT models. The first one is a non-variational model, the vanilla **Transformer** [112]. The second one is a variational model, **T-CVAE** [116] with the speculative sampling described in Section 5.4.1. Note that their hyperparameters were the same as those described in Section 3.4.2.

In addition, not only variational models but also decoding methods such as sampling-based decodings can make the outputs diverse. We employed several decoding methods and evaluated the combination. The decoding methods are as follows.

Greedy decoding chooses the output token with the highest probability in the estimated probability distribution for each time step. In Transformer, this method generates only one output. In T-CVAE, we apply greedy decoding for each sampled latent variable and it generates multiple outputs.

Beam search explores *n*-best paths of output sequences. We set the beam size to five, the same as the number of outputs.

Pure sampling randomly samples the next token according to the estimated probability distribution.

top-*k* sampling randomly samples the output token according to the estimated probability distribution. The distribution is truncated; at each time step, the next token is sampled only from the top k possible next tokens [23]. We tested the method with $k = 40$ and $k = 640$, following Holtzman et al. [36].

Nucleus sampling randomly samples the output token according to the estimated probability distribution. The distribution is truncated similarly to the top-k sampling. In this method, the candidates of possible next tokens are defined by selecting the highest probability tokens of which cumulative probability mass exceeds the pre-defined threshold p [36]. We set $p = 0.95$, following the original hyperparameter.

5.5.2 Dataset and Preprocessing

Similarly to the experiments in Section 3.4, we trained subword-based NMT models for En→Ja machine translation from the Japanese-English Subtitle Corpus

	Max. BLEU	self-BLEU
non-variational (#outputs=1)		
Transformer + Greedy decoding	14.74	-
Transformer + Beam search (beam size=5)	15.40	-
non-variational (#outputs=5)		
Transformer + Beam search (beam size=5)	22.49	72.13
Transformer + Pure sampling	13.50	12.58
Transformer + top-*k* sampling (*k*=40)	17.97	25.76
Transformer + top-*k* sampling (*k*=640)	16.84	19.84
Transformer + Nucleus sampling (*p*=0.95)	14.86	15.43
variational (#outputs=5)		
T-CVAE + Spec. sampling (*k*=5) + Greedy decoding	20.67	57.79
T-CVAE + Spec. sampling (*k*=5) + Beam search (beam size=5)	21.82	62.32
T-CVAE + Spec. sampling (*k*=5) + top-*k* sampling (*k*=40)	17.50	22.73
T-CVAE + Spec. sampling (*k*=5) + Nucleus sampling (*p*=0.95)	13.63	13.66

Table 5.8: BLEU and self-BLEU scores to generated outputs. Low self-BLEU scores indicate high diversity of model.

(JESC) [86] with the same preprocessings. As the JESC dataset was created from movie subtitles, the diversity of outputs is high and translations are likely to fail due to the subdomains of data.

5.5.3 Evaluation Metrics

Assuming a situation where a machine translation system provides several output candidates and users choose the preferred one, we evaluated the quality and diversity of five outputs generated by compared models. For evaluation of the quality, we employed BLEU [84]. Concretely, we computed sentence-BLEU scores for each output to an input and took the maximum since at least one of the generated outputs should be similar to the reference. We used NLTK 3.5 for the computation of sentence-BLEU.[8] For evaluation of the diversity, we employed self-BLEU [131] following Holtzman et al. [36]. In the metric, we compute and take the average of BLEU scores between an output and the rest among multiple generated outputs. In other words, this metric assesses how one output resembles the rest; a low self-BLEU score indicates the high diversity of a model.

[8] https://www.nltk.org/_modules/nltk/translate/bleu_score

5.5.4 Results

Table 5.8 shows the BLEU and self-BLEU scores of the compared models. First, employing sampling-based decoding made the outputs notably diverse. However, compared to **Transformer + Greedy decoding** that generated only one output sentence, the increase of the maximum BLEU score was modest. This result means that even though the models generated various outputs, they tended to be somewhat noisy and less similar to the reference. **T-CVAE + Speculative sampling (k=5) + Greedy decoding** achieved a relatively high BLEU score while the outputs were diversified. However, compared to **Transformer + Beam search (beam size=5)**, the increase made by generating multiple outputs was still lower. Combining the T-CVAE and sampling-based decoding made similar results to the Transformers with the sampling-based decoding methods. Overall, the results indicated the trade-off between the diversity of outputs and the similarity to the reference.

Although the maximum BLEU scores and the average self-BLEU scores quantitatively show the overall tendency, it is not clear how large the impact is. Additionally, we should mention that the evaluation using the maximum BLEU score does not guarantee that all generated outputs are suitable to inputs; it is possible that generated outputs with low BLEU scores can also be suitable. However, it is difficult to evaluate such outputs automatically, and conducting human evaluations for all the outputs is too costly. Thus, we show examples of generated outputs and qualitatively analyze the outputs of several models in Table 5.9.

Comparing the outputs of **Transformer + Beam search (beam size=5)** and **T-CVAE + Speculative sampling (k=5) + Greedy decoding**, the styles of the former were consistent among the outputs; for example, as for the word "definitely" in the first example, the former always translated it as "きっと" while the latter translated it as "きっと," "絶対" and "必ず." In the second example, the latter translated "work" as "研究," "仕事," and "作品." Note that all of these translations are correct when there are no constraints. Overall, in the former model, the outputs tended to be biased to the most common pattern in the dataset. In the latter model, many probable patterns depending on the subdomains appeared in the outputs when using the dataset with relatively high output diversity. As shown in Table 5.8, this tendency could lead to a slight decrease of BLEU scores by trying to generate less common patterns. However, employing variational models achieved a promising performance also in machine translation. Also, in **T-CVAE + Speculative sampling (k=5) + Beam search (beam size=5)**, applying beam search made the outputs slightly less diverse and more similar to the reference. However, the overall tendency was the same as **T-CVAE + Speculative sampling (k=5) + Greedy decoding**.

On the other hand, the outputs of Transformers with sampling-based decoding methods tended to be noisy while being diverse. Not only grammatical errors, they sometimes produced unrelated topic-words. For example, in the first example, **Transformer + top-*k* sampling (*k*=640)** chose "doctor" instead of "nurse." in the output "きっといいドクターになっちゃうからね (I'll definitely become a good doctor.)" It is reasonable that a wrong expression could sometimes be generated depending on the randomness. Although adjusting the hyperparameter k or p could mitigate the emergence of noisy expressions in return for the diversity, it is difficult to define them automatically. At least, even with the smaller k, we observed **Transformer + top-k sampling (k=40)** still tended to generate noisy outputs.

5.6 Chapter Summary

In this study, we aimed to solve KL-vanishing and to handle implicit domains in latent variables. Fine-grained domains we targeted in the thesis can significantly affect text generation while they are not necessarily available as a discrete label or a text. To model such fine-grained domains as randomness in dialogue models, we proposed a method called *speculative sampling*. This is for modulating the latent space of variational models by sampling multiple latent variables and adopting only the most probable latent variable for optimization. Although our method is quite simple and easily applicable to any variational architectures, experimental results showed that our proposed method improved the diversity of outputs while keeping the relevance to the context.

Example 1

Input	you 'll definitely become a good nurse .
Reference	あんたきっといい看護師になるよ。

Transformer + Beam search (beam size=5)

- きっといい看護師になってくれるよ。
- きっといいナースになってくれるよ。
- きっといい看護師になってくれる。
- きっといいナースになってくれるから。
- きっといいナースになってくれる。

Transformer + top-*k* sampling (*k*=40)

- お前はきっといいナースになってくれる。
- あなたは絶対に良いナースになって下さい。
- いい看護師になってあげたら絶対にいい子にちゃうからね。
- アンタが良いナースになるのって絶対にね...
- きっと私に相応しい看護師になれてしまうよ。

Transformer + top-*k* sampling (*k*=640)

- きっといいドクターになっちゃうからね
- あなたもきっといい看護婦になって
- ((とてもいい看護師になってるよ間違いない))
- パパはきっとねえいい看護師になるよ
- はぁ....いいナースになってくれるよな。こんなのって。

Transformer + Nucleus sampling (*p*=0.95)

- あなたが優秀な看護師になってくれる箇所が僕らの心に響く
- 絶対真面目になってくれるんだよ看護士さんうんだもんな。
- 春香はきっといいナースになるよ康介も。鹿かな?
- 亜美さんならきっとカッとなりますよ。
- 梨々きっといいナースになってくれ輝かしいよ。

T-CVAE + Speculative sampling (*k*=5) + Greedy decoding

- きっといい看護師になれるわ
- 絶対いい看護師になる。
- 君は必ずいい看護師になる
- いいナースになるよ
- きっといい看護師になる

T-CVAE + Speculative sampling (*k*=5) + Beam search (beam size=5)

- 絶対にいい看護師になる。
- 絶対いい看護師さんになる。
- 君は必ずいい看護師になる
- 絶対いいナースになるよ。
- きっといい看護師になるよ

Example 2	
Input	your work has been very impressive so far .
Reference	あなた の 仕事 が とても 印象 的 で し た
Transformer + Beam search (beam size=5)	
	- あなた の 仕事 は 今 の ところ 非常 に 印象 的 で し た - 君 の 仕事 は 今 の ところ 非常 に 印象 的 だっ た - あなた の 仕事 は 今 の ところ 非常 に 印象 的 だっ た わ - あなた の 仕事 は 今 の ところ 非常 に 印象 的 だっ た - 君 の 仕事 は 今 の ところ 非常 に 印象 的 だ
Transformer + top-*k* sampling (*k*=40)	
	- あなた の 作品 に は 目 を 見張 ら れ て い る わ - ここ まで の あなた の 仕事 は 非常 に 良い もの だっ た わ - 研究 で は これ まで の 成果 は 非常 に 感銘 を 受け て い ま す - 今 の ところ 君 の 調査 の 成果 は とても 優秀 だ 。 - 《 君 の 作品 よく ぞ 今 まで いっぱい こ られ た な 》
Transformer + top-*k* sampling (*k*=640)	
	- これ まで あなた の 動き は 素晴らし い もの で し た - あなた の 仕事 が 本当 に 素晴らし かっ た - [テレビ] 今日 まで の あなた の 作品 は 見事 だっ た わ 。 - お前 の 動き が 非常 に 影響 を 受け て い る - 鵜飼 さん の 勝ち で す 。
Transformer + Nucleus sampling (*p*=0.95)	
	- 貴方 の 仕事 は 今 の 所 非 常 に 印象 的 で し た - ここ まで ご 苦労 さ れ て ウィンタース さん で し た ね 。 - 君 は よく や っ て い た よ 。 - クソ 君 の 運転 の 調子 は 今 の ところ すごい - この 登山 口 に 慎重 だっ た こと 坊 も
T-CVAE + Speculative sampling (*k*=5) + Greedy decoding	
	- 今 まで の あなた の 研究 は 素晴らし い - ここ まで の 仕事 は 素晴らし い - 今 の ところ 、 あなた の 作品 は 素晴らし い で す - 君 の 作品 は 素晴らし い - 今 まで の あなた の 仕事 は 素晴らし い
T-CVAE + Speculative sampling (*k*=5) + Beam search (beam size=5)	
	- 今 まで の あなた の 研究 は 素晴らし かっ た - とても 素晴らし い 作品 だ - 今 まで の あなた の 仕事 は 素晴らし かっ た - ここ まで の あなた の 仕事 は 素晴らし かっ た 。 - 今 まで の あなた の 仕事 は 見事 だっ た

Table 5.9: Examples of generated outputs for En→Ja machine translation in JESC.

Chapter 6

Conclusions and Future Work

As discussed in Chapter 1, the difference in domains is a barrier that hinders text generation from being put into practical use. Although many existing studies have explored methods to resolve domain differences, recent neural-based methods mainly focused on coarse-grained domain differences represented as the difference in datasets. In this thesis, we expanded the target of domain adaptation and presented a multi-faceted approach for handling fine-grained domain differences that have tended to be ignored in existing studies. We consider that **1) domain-specific words and meanings** and **2) domain-specific input-output correspondences** are the crucial problems that can lead to the decrease in the performance of non-domain-aware models. In the following, we summarize our three attempts to the issues and future work. We believe our efforts in this thesis will provide a future direction of domain adaptation for the practical use of text generation.

6.1 Handling Domain-specific Words and Meanings

Related to the first problem, we proposed **vocabulary adaptation**. Although many solutions have been proposed for unknown words, the data-scarcity still makes it challenging to handle unknown words in specialized domains. Existing attempts at domain adaptation for text generation have not paid much attention to the differences in the vocabulary, the meanings of words, and the tokenization between domains. One of the reasons is the training scheme of existing neural-based models; they often build static vocabularies before training from out-domain parallel data. There has been no method to change the vocabularies themselves directly. As in-domain parallel data is assumed to be small when we need domain adaptation, it was difficult to learn the domain-specific meanings of words through the task jointly. Thus, it has been unclear how much these differences across domains affect the performance of models. To address the problem, we proposed a method to directly adapt the embedding layers of a trained model by using the embeddings pre-trained from in-domain monolingual corpora. Concretely, the pre-trained in-domain embeddings are projected into the embedding space of an out-domain NMT model for providing in-domain knowledge. We experimentally confirmed that our simple method notably improved BLEU scores for both En→Ja translation and De→En translation. Finally, although we compared several existing methods for embedding projections, there is room for exploration in the cross-domain embedding space. We plan to conduct more detailed analyses of the embedding space and improve the projection method.

6.2 Explicit Modeling of Domain-specific Input-output Correspondence

For the second problem, we proposed **situation-aware models**. We commonly collect data that is relatively similar to the target domain and regard the whole dataset as one domain. However, situations or fine-grained domains are different from each other even among examples in the same data, which can inevitably affect outputs of text generation models. For instance, we converse with others while (implicitly) considering the utterance and other various conversational situations (Section 3.3) such as time, place, and the current context of a conversation, and even our relationship with the addressee. We aimed to explicitly introduce such situations into text generation models as influential factors of the probable outputs to a given input. Concretely, we proposed local-global SEQ2SEQ and SEQ2SEQ with situation embeddings while targeting the user, time, and topic of a conversation as situations.

In this attempt, for investigating the effects of situations in end-to-end text generation separately from the costs to obtain the situations and their accuracies, we focused on the situations that were relatively easy to obtain. To make more clues available in text generation, we believe that it is necessary as future work to explore techniques for estimation of various situations or methods for implicitly utilizing the situation like the approach described in the next section.

6.3 Implicit Modeling of Domain-specific Input-output Correspondence

As another approach to address the second problem, we proposed **speculative sampling** for variational models that are a promising method to model implicit domains. As shown in Chapter 4, fine-grained domains such as situations we targeted can significantly affect text generation. However, such domains are not necessarily available; they may not be publicly available due to privacy concerns or not recorded as explicit labels. We considered that such inaccessible domains could also involve the output of models and aimed to capture them in the latent space of variational models implicitly. The problem we address in this work is the discrepancy between a sampled latent variable and a training example. Even under the observation of the ground-truth response to a given utterance, the latent variable sampled from the posterior distribution of a model can be inappropriate for the response, making the latent space of the model disorganized. We aimed to resolve the problem by speculatively sampling latent variables and adopting only the most probable and promising latent variable for optimization. Although our proposed method is simple and easily applicable, experimental results showed improved diversity in generated responses while it did not lose relevance to the context. In the experiments, we employed a standard variational model that hypothesis a multivariate Gaussian distribution for the latent space. However, existing studies suggested that suitable distributions of variational models may vary depending on the nature of data. We plan to conduct a further investigation about how the distributions and the methods for training and decoding can affect text generation as future work.

6.4 Other Research Activities in Doctoral Course

Besides the three primary research efforts introduced in the thesis, I briefly mention the overview of representative studies in the activities where I worked as a co-author and their relation to the main researches [81, 82, 78–80, 38].

Modeling and analyzing personal biases in NN-based representations [78–80]: In the trials for situation-aware dialogue modeling introduced in Chapter 4, we considered personalized representations had room for being explored. Particularly, personalized representations do not necessarily contain helpful information (e.g., demographic factors or preferences of writers). Instead, personalized representations learned through a task can contain task-specific biases. For example, in text classification tasks, employing word representations personalized to a writer can directly increase the probabilities of outputs that the writer preferred in the training data regardless of the nuances in word meanings. We considered that such biases make both analyzing and transferring the representations difficult, and proposed a method for debiasing the personalized representations.

Dialogue agents autonomously inquiring missing information [81]: Related to the motivation of the study described in Chapter 5, all fine-grained situations of conversations are not available. However, it is unnecessary to have all such situations before starting a conversation; even in human-to-human conversations, we try to know the information about conversation partners by asking questions or by the conversation itself. Analogically, we aimed to develop dialogue agents that can supplement the important information through questions.

Describing the meanings of unknown or infrequent terms [38]: Related to the study introduced in Chapter 3, we attempted another approach to understanding and handling unknown or infrequent terms. In the study of vocabulary adaptation, we hypothesized that it is relatively easier to obtain in-domain monolingual data than in-domain parallel data and tried to transfer in-domain vocabularies and embeddings to out-domain NMT models. However, even in-domain monolingual data or a knowledge-base describing the unknown words are not necessarily available. In such situations, it is inevitable to infer the meaning from contexts where the unknown words appear. For this reason, we explored methods to obtain the meaning of words as a description from contexts and investigated to what extent existing NN-based representations capture the meanings of words.

Bibliography

[1] Adiwardana, D., Luong, M.-T., So, D. R., Hall, J., Fiedel, N., Thoppilan, R., Yang, Z., Kulshreshtha, A., Nemade, G., Lu, Y., et al. (2020). Towards a human-like open-domain chatbot. *arXiv preprint arXiv:2001.09977*.

[2] Aji, A. F., Bogoychev, N., Heafield, K., and Sennrich, R. (2020). In neural machine translation, what does transfer learning transfer? In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL 2020)*, pages 7701–7710.

[3] Ba, J. L., Kiros, J. R., and Hinton, G. E. (2016). Layer normalization. *arXiv preprint arXiv:1607.06450*.

[4] Bahdanau, D., Cho, K., and Bengio, Y. (2015). Neural machine translation by jointly learning to align and translate. In *Proceedings of the third International Conference on Learning Representations (ICLR 2015)*.

[5] Baheti, A., Ritter, A., Li, J., and Dolan, B. (2018). Generating more interesting responses in neural conversation models with distributional constraints. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP 2018)*, pages 3970–3980.

[6] Bak, J. and Oh, A. (2019). Variational hierarchical user-based conversation model. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP 2019)*, pages 1941–1950.

[7] Bamman, D., O'Connor, B., and Smith, N. A. (2013). Learning latent personas of film characters. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers) (ACL 2013)*, pages 352–361.

[8] Bapna, A. and Firat, O. (2019). Simple, scalable adaptation for neural machine translation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP 2019)*, pages 1538–1548.

[9] Bowman, S. R., Vilnis, L., Vinyals, O., Dai, A., Jozefowicz, R., and Bengio, S. (2016). Generating sentences from a continuous space. In *Proceedings of The*

20th SIGNLL Conference on Computational Natural Language Learning (CoNLL 2016), pages 10–21.

[10] Britz, D., Le, Q., and Pryzant, R. (2017). Effective domain mixing for neural machine translation. In *Proceedings of the Second Conference on Machine Translation (WMT 2017)*, pages 118–126.

[11] Brown, P. F., Della Pietra, S. A., Della Pietra, V. J., and Mercer, R. L. (1993). The mathematics of statistical machine translation: Parameter estimation. *Computational Linguistics*, 19(2):263–311.

[12] Budzianowski, P., Wen, T.-H., Tseng, B.-H., Casanueva, I., Ultes, S., Ramadan, O., and Gašić, M. (2018). MultiWOZ - a large-scale multi-domain Wizard-of-Oz dataset for task-oriented dialogue modelling. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP 2020)*, pages 5016–5026.

[13] Chen, L., Lv, B., Wang, C., Zhu, S., Tan, B., and Yu, K. (2020). Schema-guided multi-domain dialogue state tracking with graph attention neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI 2020)*, volume 34, pages 7521–7528.

[14] Chen, W., Chen, J., Qin, P., Yan, X., and Wang, W. Y. (2019). Semantically conditioned dialog response generation via hierarchical disentangled self-attention. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (ACL 2019)*, pages 3696–3709.

[15] Cho, K., van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., and Bengio, Y. (2014). 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 2014)*, pages 1724–1734.

[16] Chu, C., Dabre, R., and Kurohashi, S. (2017). An empirical comparison of domain adaptation methods for neural machine translation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers) (ACL 2017)*, pages 385–391.

[17] Chu, C. and Wang, R. (2018). A survey of domain adaptation for neural machine translation. In *Proceedings of the 27th International Conference on Computational Linguistics (COLING 2018)*, pages 1304–1319.

[18] Chu, E., Vijayaraghavan, P., and Roy, D. (2018). Learning personas from dialogue with attentive memory networks. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP 2018)*, pages 2638–2646. Association for Computational Linguistics.

[19] Daumé III, H. (2007). Frustratingly easy domain adaptation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics (ACL 2007)*, pages 256–263.

[20] Dinu, G., Mathur, P., Federico, M., and Al-Onaizan, Y. (2019). Training neural machine translation to apply terminology constraints. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (ACL 2019)*, pages 3063–3068.

[21] Domhan, T. and Hieber, F. (2017). Using target-side monolingual data for neural machine translation through multi-task learning. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing (EMNLP 2017)*, pages 1500–1505.

[22] Elman, J. L. (1990). Finding structure in time. *Cognitive science*, 14(2):179–211.

[23] Fan, A., Lewis, M., and Dauphin, Y. (2018). Hierarchical neural story generation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers) (ACL 2018)*, pages 889–898.

[24] Freitag, M. and Al-Onaizan, Y. (2016). Fast domain adaptation for neural machine translation. *CoRR*, abs/1612.06897.

[25] Fu, H., Li, C., Liu, X., Gao, J., Celikyilmaz, A., and Carin, L. (2019). Cyclical annealing schedule: A simple approach to mitigating KL vanishing. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers) (NAACL 2019)*, pages 240–250.

[26] Gao, S., Sethi, A., Agarwal, S., Chung, T., and Hakkani-Tur, D. (2019a). Dialog state tracking: A neural reading comprehension approach. In *Proceedings of the 20th Annual SIGdial Meeting on Discourse and Dialogue (SIGDIAL 2019)*, pages 264–273.

[27] Gao, X., Lee, S., Zhang, Y., Brockett, C., Galley, M., Gao, J., and Dolan, B. (2019b). Jointly optimizing diversity and relevance in neural response generation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers) (NAACL-HLT 2019)*, pages 1229–1238.

[28] Gao, X., Zhang, Y., Lee, S., Galley, M., Brockett, C., Gao, J., and Dolan, B. (2019c). Structuring latent spaces for stylized response generation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP 2019)*, pages 1814–1823.

[29] Goldberg, Y. and Nivre, J. (2012). A dynamic oracle for arc-eager dependency parsing. In *Proceedings of the 24th International Conference on Computational Linguistics (COLING 2012)*, pages 959–976.

[30] Gu, S., Feng, Y., and Liu, Q. (2019). Improving domain adaptation translation with domain invariant and specific information. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (NAACL 2019)*, pages 3081–3091.

[31] Gu, X., Cho, K., Ha, J.-W., and Kim, S. (2018). Dialogwae: Multimodal response generation with conditional wasserstein auto-encoder. *arXiv preprint arXiv:1805.12352*.

[32] Gülçehre, Ç., Firat, O., Xu, K., Cho, K., Barrault, L., Lin, H., Bougares, F., Schwenk, H., and Bengio, Y. (2015). On using monolingual corpora in neural machine translation. *CoRR*, abs/1503.03535.

[33] Gupta, P., Schütze, H., and Andrassy, B. (2016). Table filling multi-task recurrent neural network for joint entity and relation extraction. In *Proceedings of the 26th International Conference on Computational Linguistics (COLING 2016)*, pages 2537–2547.

[34] Hasegawa, T., Kaji, N., Yoshinaga, N., and Toyoda, M. (2013). Predicting and eliciting addressee's emotion in online dialogue. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (ACL 2013)*, pages 964–972.

[35] Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, 9(8):1735–1780.

[36] Holtzman, A., Buys, J., Du, L., Forbes, M., and Choi, Y. (2020). The curious case of neural text degeneration.

[37] Hu, J., Xia, M., Neubig, G., and Carbonell, J. (2019). Domain adaptation of neural machine translation by lexicon induction. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (ACL 2019)*, pages 2989–3001.

[38] Ishiwatari, S., Hayashi, H., Yoshinaga, N., Neubig, G., Sato, S., Toyoda, M., and Kitsuregawa, M. (2019). Learning to describe unknown phrases with local and global contexts. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers) (NAACL 2019)*, pages 3467–3476.

[39] Jean, S., Cho, K., Memisevic, R., and Bengio, Y. (2015). On using very large target vocabulary for neural machine translation. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th*

International Joint Conference on Natural Language Processing (ACL-IJCNLP 2015), pages 1–10.

[40] Johnson, M., Schuster, M., Le, Q. V., Krikun, M., Wu, Y., Chen, Z., Thorat, N., Viégas, F., Wattenberg, M., Corrado, G., et al. (2016). Google's multilingual neural machine translation system: Enabling zero-shot translation. *arXiv preprint arXiv:1611.04558*.

[41] Khanpour, H., Guntakandla, N., and Nielsen, R. (2016). Dialogue act classification in domain-independent conversations using a deep recurrent neural network. In *Proceedings of the 26th International Conference on Computational Linguistics: Technical Papers (COLING 2016)*, pages 2012–2021.

[42] Khayrallah, H., Thompson, B., Duh, K., and Koehn, P. (2018). Regularized training objective for continued training for domain adaptation in neural machine translation. In *Proceedings of the 2nd Workshop on Neural Machine Translation and Generation (WNMT 2018)*, pages 36–44.

[43] Kim, Y.-B., Stratos, K., and Sarikaya, R. (2016). Frustratingly easy neural domain adaptation. In *Proceedings of the 26th International Conference on Computational Linguistics: Technical Papers (COLING 2016)*, pages 387–396.

[44] Kingma, D. and Ba, J. (2015). Adam: A method for stochastic optimization. In *Proceedings of the third International Conference on Learning Representations (ICLR 2015)*.

[45] Kingma, D. P., Salimans, T., Jozefowicz, R., Chen, X., Sutskever, I., and Welling, M. (2016). Improved variational inference with inverse autoregressive flow. In *Advances in neural information processing systems (NIPS 2016)*, pages 4743–4751.

[46] Kingma, D. P. and Welling, M. (2014). Auto-encoding variational bayes.

[47] Kobori, T., Nakano, M., and Nakamura, T. (2016). Small talk improves user impressions of interview dialogue systems. In *Proceedings of the 17th Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDIAL 2016)*, pages 370–380.

[48] Kobus, C., Crego, J., and Senellart, J. (2017). Domain control for neural machine translation. In *Proceedings of the International Conference Recent Advances in Natural Language Processing, (RANLP 2017)*, pages 372–378.

[49] Koehn, P. and Knowles, R. (2017). Six challenges for neural machine translation. In *Proceedings of the First Workshop on Neural Machine Translation (WNMT 2017)*, pages 28–39.

[50] Kruengkrai, C. (2019). Better exploiting latent variables in text modeling. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (ACL 2019)*, pages 5527–5532.

[51] Kudo, T. (2018). Subword regularization: Improving neural network translation models with multiple subword candidates. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (ACL 2018)*, pages 66–75.

[52] Kudo, T. and Richardson, J. (2018). SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations (EMNLP 2018)*, pages 66–71.

[53] Li, J., Galley, M., Brockett, C., Gao, J., and Dolan, B. (2016a). A diversity-promoting objective function for neural conversation models. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT 2016)*, pages 110–119.

[54] Li, J., Galley, M., Brockett, C., Spithourakis, G. P., Gao, J., and Dolan, B. (2016b). A persona-based neural conversation model. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (ACL 2016)*, pages 994–1003.

[55] Li, J., Luo, P., Lin, F., and Chen, B. (2018). Conversational model adaptation via KL divergence regularization. In *Proceedigs of the 32nd AAAI Conference on Artificial Intelligence (AAAI 2018)*, pages 5213–5219.

[56] Ling, W., Trancoso, I., Dyer, C., and Black, A. W. (2015). Character-based neural machine translation. *CoRR*, abs/1511.04586.

[57] Liu, C.-W., Lowe, R., Serban, I., Noseworthy, M., Charlin, L., and Pineau, J. (2016a). How not to evaluate your dialogue system: An empirical study of unsupervised evaluation metrics for dialogue response generation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing (EMNLP 2016)*, pages 2122–2132.

[58] Liu, P., Qiu, X., and Huang, X. (2016b). Deep multi-task learning with shared memory for text classification. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing (EMNLP 2016)*, pages 118–127.

[59] Liu, P., Qiu, X., and Huang, X. (2016c). Recurrent neural network for text classification with multi-task learning. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence (IJCAI 2016)*, pages 2873–2879.

[60] Luong, M.-T., Le, Q. V., Sutskever, I., Vinyals, O., and Kaiser, L. (2016). Multi-task sequence to sequence learning. In *Proceedings of the fifth International Conference on Learning Representations (ICLR 2016)*.

[61] Luong, M.-T. and Manning, C. D. (2015). Stanford neural machine translation systems for spoken language domains. In *Proceedings of the International Workshop on Spoken Language Translation (IWSLT 2015)*, pages 76–79.

[62] Luong, M.-T. and Manning, C. D. (2016). Achieving open vocabulary neural machine translation with hybrid word-character models. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (ACL 2016)*, pages 1054–1063.

[63] Luong, T., Pham, H., and Manning, C. D. (2015a). Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP 2015)*, pages 1412–1421.

[64] Luong, T., Sutskever, I., Le, Q., Vinyals, O., and Zaremba, W. (2015b). Addressing the rare word problem in neural machine translation. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (ACL-IJCNLP 2015)*, pages 11–19.

[65] Ma, X. and Hovy, E. (2016). End-to-end sequence labeling via bi-directional LSTM-CNNs-CRF. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers) (ACL 2016)*, pages 1064–1074.

[66] Maaten, L. v. d. and Hinton, G. (2008). Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(Nov):2579–2605.

[67] Mathur, P., Venkatapathy, S., and Cancedda, N. (2014). Fast domain adaptation of smt models without in-domain parallel data. In *Proceedings of COLING*, pages 1114–1123.

[68] Mazare, P.-E., Humeau, S., Raison, M., and Bordes, A. (2018). Training millions of personalized dialogue agents. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP 2018)*, pages 2775–2779.

[69] Michel, P. and Neubig, G. (2018). Extreme adaptation for personalized neural machine translation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers) (ACL 2018, short)*, pages 312–318.

[70] Mikolov, T., Karafiát, M., Burget, L., Cernockỳ, J., and Khudanpur, S. (2010). Recurrent neural network based language model. In *Proceedings of INTERSPEECH*, volume 2, page 3.

[71] Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., and Dean, J. (2013). Distributed representations of words and phrases and their compositionality.

In *Proceedings of the 26th International Conference on Neural Information Processing Systems (NIPS 2013)*, pages 3111–3119.

[72] Mirkin, S. and Meunier, J.-L. (2015). Personalized machine translation: Predicting translational preferences. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP 2015)*, pages 2019–2025, Lisbon, Portugal.

[73] Nakamura, R., Sudoh, K., Yoshino, K., and Nakamura, S. (2019). Another diversity-promoting objective function for neural dialogue generation. In *Proceedings of the second AAAI 2019 Workshop on Reasoning and Learning for Human-Machine Dialogues (DEEP-DIAL 2019)*.

[74] Nakazawa, T., Yaguchi, M., Uchimoto, K., Utiyama, M., Sumita, E., Kurohashi, S., and Isahara, H. (2016). ASPEC: Asian scientific paper excerpt corpus. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016)*, pages 2204–2208.

[75] Neishi, M., Sakuma, J., Tohda, S., Ishiwatari, S., Yoshinaga, N., and Toyoda, M. (2017). A bag of useful tricks for practical neural machine translation: Embedding layer initialization and large batch size. In *Proceedings of the 4th Workshop on Asian Translation (WAT 2017)*, pages 99–109.

[76] Ni, K. and Wang, W. Y. (2017). Learning to explain non-standard English words and phrases. In *Proceedings of the 8th International Joint Conference on Natural Language Processing (IJCNLP 2017, short papers)*, pages 413–417.

[77] Nisioi, S., Štajner, S., Ponzetto, S. P., and Dinu, L. P. (2017). Exploring neural text simplification models. In *Proceedings of the 55th annual meeting of the Association for Computational Linguistics (ACL 2017, short papers)*, pages 85–91.

[78] Oba, D., Sato, S., Akasaki, S., Yoshinaga, N., and Toyoda, M. (2020). Personal semantic variations in word meanings: Induction, application, and analysis. *Journal of Natural Language Processing*, 27(2):467–490.

[79] Oba, D., Sato, S., Yoshinaga, N., Akasaki, S., and Toyoda, M. (2019a). Understanding interpersonal variations in word meanings via review target identification. In *Proceedings of the 20th International Conference on Computational Linguistics and Intelligent Text Processing (CICLing 2019)*, number 129.

[80] Oba, D., Yoshinaga, N., Sato, S., Akasaki, S., and Toyoda, M. (2019b). Modeling personal biases in language use by inducing personalized word embeddings. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers) (NAACL-HLT 2019, short)*, pages 2102–2108.

[81] Ohara, K., Sato, S., Yoshinaga, N., Toyoda, M., and Kitsuregawa, M. (2017). Detecting necessity of questioning towards a dialogue agent autonomously inquiring missing information. In *Proceedings of the 9th forum on Data Engineering and Information Management (DEIM 2017, in Japanese)*.

[82] Ohara, K., Sato, S., Yoshinaga, N., Toyoda, M., and Kitsuregawa, M. (2018). Predicting dialogue acts of responses in online dialogue using hierarchical rnn. In *Proceedings of the 24th Annual Meeting of the Association for Natural Language Processing (NLP 2018, in Japanese)*.

[83] Ott, M., Edunov, S., Baevski, A., Fan, A., Gross, S., Ng, N., Grangier, D., and Auli, M. (2019). fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations) (NAACL 2019)*, pages 48–53.

[84] Papineni, K., Roukos, S., Ward, T., and Zhu, W.-J. (2002). Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL 2002)*, pages 311–318.

[85] Poerner, N., Waltinger, U., and Schütze, H. (2020). Inexpensive domain adaptation of pretrained language models: Case studies on biomedical NER and covid-19 QA. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings (EMNLP 2020: Findings)*, pages 1482–1490.

[86] Pryzant, R., Chung, Y., Jurafsky, D., and Britz, D. (2018). JESC: Japanese-English subtitle corpus. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*.

[87] Rabinovich, E., Patel, R. N., Mirkin, S., Specia, L., and Wintner, S. (2017). Personalized machine translation: Preserving original author traits. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers (EACL 2017)*, pages 1074–1084.

[88] Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., and Liu, P. J. (2019). Exploring the limits of transfer learning with a unified text-to-text transformer. *CoRR*, abs/1910.10683.

[89] Ramadan, O., Budzianowski, P., and Gašić, M. (2018). Large-scale multi-domain belief tracking with knowledge sharing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers) (ACL 2018, short)*, pages 432–437.

[90] Reiter, R. (1981). On closed world data bases. In *Readings in artificial intelligence*, pages 119–140. Elsevier.

[91] Ren, L., Ni, J., and McAuley, J. (2019). Scalable and accurate dialogue state tracking via hierarchical sequence generation. *arXiv preprint arXiv:1909.00754*.

[92] Ritter, A., Cherry, C., and Dolan, W. B. (2011). Data-driven response generation in social media. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing (EMNLP-11)*, pages 583–593.

[93] Sakuma, J. and Yoshinaga, N. (2019). Multilingual model using cross-task embedding projection. In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL 2019)*, pages 22–32.

[94] Sato, S., Sakuma, J., Yoshinaga, N., Toyoda, M., and Kitsuregawa, M. (2020). Vocabulary adaptation for domain adaptation in neural machine translation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings (EMNLP 2020: Findings)*, pages 4269–4279.

[95] Sato, S., Shonosuke Ishiwatari, N. Y., Toyoda, M., and Kitsuregawa, M. (2016). UT dialogue system at NTCIR-12 STC. In *Proceedings of the 12th NTCIR Conference on Evaluation of Information Access Technologies*, pages 518–522.

[96] Sato, S., Yoshinaga, N., Toyoda, M., and Kitsuregawa, M. (2017). Modeling situations in neural chat bots. In *Proceedings of ACL 2017, Student Research Workshop*, pages 120–127.

[97] Sennrich, R., Haddow, B., and Birch, A. (2016a). Improving neural machine translation models with monolingual data. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (ACL 2016)*, pages 86–96.

[98] Sennrich, R., Haddow, B., and Birch, A. (2016b). Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (ACL 2016)*, pages 1715–1725.

[99] Sennrich, R. and Zhang, B. (2019). Revisiting low-resource neural machine translation: A case study. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (ACL 2019)*, pages 211–221.

[100] Serban, I., Sordoni, A., Lowe, R., Charlin, L., Pineau, J., Courville, A., and Bengio, Y. (2017). A hierarchical latent variable encoder-decoder model for generating dialogues.

[101] Serban, I. V., Sordoni, A., Bengio, Y., Courville, A., and Pineau, J. (2016). Building end-to-end dialogue systems using generative hierarchical neural network models. In *Proceedigs of the 30th AAAI Conference on Artificial Intelligence (AAAI 2016)*.

[102] Shang, L., Lu, Z., and Li, H. (2015). Neural responding machine for short-text conversation. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (ACL-IJCNLP 2015)*, pages 1577–1586.

[103] Shen, X., Su, H., Niu, S., and Demberg, V. (2018). Improving variational encoder-decoders in dialogue generation. In *Proceedings of 32nd AAAI Conference on Artificial Intelligence (AAAI 2018)*, pages 5456—-5463.

[104] Song, Z., Zheng, X., Liu, L., Xu, M., and Huang, X. (2019). Generating responses with a specific emotion in dialog. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (ACL 2019)*, pages 3685–3695.

[105] Sordoni, A., Galley, M., Auli, M., Brockett, C., Ji, Y., Mitchell, M., Nie, J.-Y., Gao, J., and Dolan, B. (2015). A neural network approach to context-sensitive generation of conversational responses. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT 2015)*, pages 196–205.

[106] Sutskever, I., Vinyals, O., and Le, Q. V. (2014). Sequence to sequence learning with neural networks. In *Proceedings of the 27th International Conference on Neural Information Processing Systems (NIPS 2014)*, pages 3104–3112.

[107] Thompson, B., Khayrallah, H., Anastasopoulos, A., McCarthy, A. D., Duh, K., Marvin, R., McNamee, P., Gwinnup, J., Anderson, T., and Koehn, P. (2018). Freezing subnetworks to analyze domain adaptation in neural machine translation. In *Proceedings of the Third Conference on Machine Translation: Research Papers (WMT 2018)*, pages 124–132.

[108] Tiedemann, J. (2012). Parallel data, tools and interfaces in OPUS. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC 2012)*, pages 2214–2218.

[109] Tran, V.-K. and Nguyen, L.-M. (2018). Adversarial domain adaptation for variational neural language generation in dialogue systems. In *Proceedings of the 27th International Conference on Computational Linguistics (COLING 2018)*, pages 1205–1217.

[110] Turing, A. M. (2009). Computing machinery and intelligence. In *Parsing the turing test*, pages 23–65. Springer.

[111] TURING, I. B. A. (1950). Computing machinery and intelligence-am turing. *Mind*, 59(236):433.

[112] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L. u., and Polosukhin, I. (2017). Attention is all you need. In Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., and Garnett, R., editors, *Advances in Neural Information Processing Systems 30 (NIPS 2017)*, pages 5998–6008.

[113] Vinyals, O., Kaiser, Ł., Koo, T., Petrov, S., Sutskever, I., and Hinton, G. (2015). Grammar as a foreign language. In *Advances in neural information processing systems (NIPS 2015)*, pages 2773–2781.

[114] Vinyals, O. and Le, Q. V. (2015). A neural conversational model. In *Proceedings of Deep Learning Workshop held at the 31st International Conference on Machine Learning (ICML 2015)*.

[115] Wang, R., Utiyama, M., Liu, L., Chen, K., and Sumita, E. (2017). Instance weighting for neural machine translation domain adaptation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing (EMNLP 2017)*, pages 1482–1488.

[116] Wang, T. and Wan, X. (2019). T-cvae: Transformer-based conditioned variational autoencoder for story completion. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence (IJCAI 2019)*, pages 5233–5239.

[117] Wen, T.-H., Gasic, M., Mrkšić, N., Su, P.-H., Vandyke, D., and Young, S. (2015). Semantically conditioned lstm-based natural language generation for spoken dialogue systems. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP 2015)*, pages 1711–1721.

[118] Wen, T.-H., Vandyke, D., Mrkšić, N., Gasic, M., Rojas Barahona, L. M., Su, P.-H., Ultes, S., and Young, S. (2017). A network-based end-to-end trainable task-oriented dialogue system. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics (EACL 2017)*, pages 438–449.

[119] Williams, J. D. and Young, S. (2007). Partially observable markov decision processes for spoken dialog systems. *Computer Speech and Language*, 21(2):393–422.

[120] Wu, B., Wang, B., and Xue, H. (2016). Ranking responses oriented to conversational relevance in chat-bots. In *Proceedings of the 26th International Conference on Computational Linguistics (COLING 2016)*, pages 652–662.

[121] Wu, C.-S., Madotto, A., Hosseini-Asl, E., Xiong, C., Socher, R., and Fung, P. (2019). Transferable multi-domain state generator for task-oriented dialogue systems. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (ACL 2019)*, pages 808–819.

[122] Wuebker, J., Simianer, P., and DeNero, J. (2018). Compact personalized models for neural machine translation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP 2018)*, pages 881–886.

[123] Xing, C., Wang, D., Liu, C., and Lin, Y. (2015). 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 (NAACL 2015), pages 1006–1011.

[124] Xing, C., Wu, W., Wu, Y., Liu, J., Huang, Y., Zhou, M., and Ma, W.-Y. (2017). Topic aware neural response generation. In *Proceedings of the 31st AAAI Conference on Artificial Intelligence (AAAI 2017)*, pages 3351–3357.

[125] Yuan, Z. and Briscoe, T. (2016). Grammatical error correction using neural machine translation. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT 2016)*, pages 380–386, San Diego, California. Association for Computational Linguistics.

[126] Zaremba, W., Sutskever, I., and Vinyals, O. (2014). Recurrent neural network regularization. In *Proceedings of the second International Conference on Learning Representations (ICLR 2014)*.

[127] Zhang, J., Hashimoto, K., Wu, C.-S., Wang, Y., Yu, P., Socher, R., and Xiong, C. (2020). Find or classify? dual strategy for slot-value predictions on multi-domain dialog state tracking. In *Proceedings of the Ninth Joint Conference on Lexical and Computational Semantics (*SEM 2020)*, pages 154–167.

[128] Zhang, S., Dinan, E., Urbanek, J., Szlam, A., Kiela, D., and Weston, J. (2018). Personalizing dialogue agents: I have a dog, do you have pets too? In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers) (ACL 2018)*, pages 2204–2213.

[129] Zhao, T., Zhao, R., and Eskenazi, M. (2017). Learning discourse-level diversity for neural dialog models using conditional variational autoencoders. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers) (ACL 2017)*, pages 654–664.

[130] Zhou, H., Huang, M., Zhang, T., Zhu, X., and Liu, B. (2018). Emotional chatting machine: Emotional conversation generation with internal and external memory. *ArXiv*, abs/1704.01074.

[131] Zhu, Y., Lu, S., Zheng, L., Guo, J., Zhang, W., Wang, J., and Yu, Y. (2018). Texygen: A benchmarking platform for text generation models. In *The 41st International ACM SIGIR Conference on Research amp; Development in Information Retrieval (SIGIR 2018)*, page 1097–1100.

Publications

Publications related to the thesis

International conference (reviewed)

[1] Shoetsu Sato, Jin Sakuma, Naoki Yoshinaga, Masashi Toyoda, Masaru Kitsuregawa. “Vocabulary Adaptation for Distant Domain Adaptation in Neural Machine Translation.” (EMNLP 2020, findings).

[2] Shoetsu Sato, Naoki Yoshinaga, Masashi Toyoda, Masaru Kitsuregawa. “Modeling Situations in Neural Chat Bots.” (ACL-SRW 2017).

International conference (non-reviewed)

[3] Shoetsu Sato, Shonosuke Ishiwatari, Naoki Yoshinaga, Masashi Toyoda, Masaru Kitsuregawa. “UT Dialogue System at NTCIR-12 STC.” Proceedings of the 12th NTCIR Conference on Evaluation of Information Access (NTCIR 2016)

Domestic conference

[4] 佐藤翔悦, 喜連川優. “潜在変数の投機的サンプリングに基づく多様な雑談応答生成”. 言語処理学会第27回年次大会 (NLP 2021)

[5] 佐藤翔悦, 佐久間仁, 吉永直樹, 豊田正史, 喜連川優．“語彙切換に基づくニューラル機械翻訳の遠ドメイン適応”．言語処理学会第26回年次大会, (NLP 2020)

[6] 佐藤翔悦，吉永直樹，石渡祥之佑，豊田正史，喜連川優, “非明示的な発話状況を考慮したニューラル対話モデルの検討”. 第31回人工知能学会全国大会 (JSAI 2017).

[7] 佐藤翔悦，吉永直樹，豊田正史，喜連川優．“暗黙の発話状況を考慮したニューラル対話モデル”. 言語処理学会第23回年次大会 (NLP 2017)

[8] 佐藤翔悦，吉永直樹，豊田正史，喜連川優．“発話状況を意識したオンライン上の対話における応答選択”. 第30回人工知能学会全国大会 (JSAI 2016)

Publications on research derived from the thesis

Journal

[9] Daisuke Oba, Shoetsu Sato, Satoshi Akasaki, Naoki Yoshinaga, Masashi Toyoda. “Personal Semantic Variations in Word Meanings: Induction, Application, and Analysis.” Journal of Natural Language Processsing, Volume 27, Number 2, June 2020

International conference

[10] Shonosuke Ishiwatari, Hiroaki Hayashi, Naoki Yoshinaga, Graham Neubig, Shoetsu Sato, Masashi Toyoda and Masaru Kitsuregawa. “Learning to Describe Unknown Phrases with Local and Global Contexts.” (NAACL 2019)

[11] Daisuke Oba, Naoki Yoshinaga, Shoetsu Sato, Satoshi Akasaki and Masashi Toyoda. “Modeling Personal Biases in Language Use by Inducing Personalized Word Embeddings.” (NAACL 2019)

[12] Daisuke Oba, Shoetsu Sato, Naoki Yoshinaga, Satoshi Akasaki, Masashi Toyoda. “Understanding Interpersonal Variations in Word Meanings via Review Target Identification.” Proceedings of the 20th International Conference on Computational Linguistics and Lntelligent Text Processing (CICLing 2019)

Domestic conference

[13] 大葉大輔, 佐藤翔悦, 赤崎智, 吉永直樹, 豊田正史. “人の言語使用における単語の意味の揺らぎの解明に向けて”. 言語処理学会第25回年次大会 (NLP 2019)

[14] 大原康平，佐藤翔悦，吉永直樹，豊田正史，喜連川優. “階層型 RNN を用いた対話における応答の対話行為予測”. 言語処理学会第24回年次大会 (NLP 2018)

[15] 大原康平，佐藤翔悦，吉永直樹，豊田正史，喜連川優. “不足情報を自律的に問う対話エージェントの実現に向けた聞き返しの必要性検知”. 第9回データ工学と情報マネジメントに関するフォーラム (DEIM 2017)

Publications non-related to the thesis

Domestic conference

[16] 福田展和, 佐藤翔悦, 吉永直樹, 喜連川優. “Wikipediaの内部リンクを用いた弱教師あり共参照解析”. 言語処理学会第25回年次大会 (NLP 2019)