Enhancing Summarization and Causal Discovery: Topic Awareness, Normalizing Flows, and Hierarchical Ensembles

A DISSERTATION
SUBMITTED TO THE FACULTY OF THE GRADUATE SCHOOL
OF THE UNIVERSITY OF MINNESOTA
BY

Yu Yang

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

Xiaotong Shen, Adviser

June 2023

© Yu Yang 2023

Acknowledgements

I am very grateful to my adviser, Xiaotong Shen. His influence on my research, intellectual growth, and professional development is immeasurable and transcends the scope of this thesis. He has been a beacon of guidance and support throughout my journey. I have always enjoyed our conversations on a wide range of interesting topics. I am also extremely appreciative of Jie Ding, Wei Pan, and Adam Joseph Rothman, who served on my PhD committee and provided valuable feedback that enhanced my research.

I wish to express my gratitude to my co-authors Sthitie Bom and Chunlin Li for their contributions and assistance. Sthitie has been consistently supportive and caring throughout our collaboration with Seagate Technology. Chunlin has been a great help in my research. Our discussions have always been intellectually stimulating, leading to significant learning experiences for me.

My peers in the statistics department and across the university have made my time in Minnesota truly enjoyable. A heartfelt thanks to Christina Zhao, Marten Thompson, Tate Jacobson, Xuesong Hou, Yash Travadi, Tuan Pham, Yuchen Yao, Yuefei Shen, Chunlin Li, Ruoyu He, Jiawei Zhang, Yifei Liu, Ganghua Wang, Xiangyu Zhang, Robert Oliver VandenBerg, Tianyi Wei, and Yifan Yan. Christina deserves an extra note of gratitude for her steady support, care, and for the countless fun moments we've shared together.

Lastly, my deepest gratitude goes to my parents. Thank you to my mom Julan and dad Xiaoqiang for your unwavering commitment and support throughout my life. I love you always.

Dedication

To my parents, Julan Lou and Xiaoqiang Yang, who give me endless love and courage to keep moving forward.

Abstract

This doctoral thesis delves into the realms of abstractive summarization and causal discovery within complex systems. I present a set of new methods that counter prevailing challenges, uncovering the significant roles that topic awareness, normalizing flows, and hierarchical ensemble techniques can play in enhancing text summarization and causal discovery, respectively.

The first part of the thesis investigates abstractive summarization, introducing PA-TAM, a model that employs a hierarchical approach to incorporate topic information at both document and sentence levels and a penalized attention mechanism to reduce textual repetitions. The application of these techniques results in the generation of coherent and informative summaries. Furthermore, I propose FlowSUM, a normalizing flows-based variational encoder-decoder framework tailored for Transformer-based summarization models. FlowSUM mitigates challenges related to capturing complex semantic structures and dealing with posterior collapse during training, thereby enriching the latent posterior distribution and improving summary quality. FlowSUM is also shown to possess great potential for transferring knowledge from large language models.

The second part of the thesis focuses on causal discovery, particularly targeting the wafer manufacturing domain. I propose a hierarchical ensemble approach that leverages temporal and domain constraints, simultaneously handling challenges such as high-dimensional, mixed, and imbalanced data, as well as irregular missing patterns. The efficacy of this approach is substantiated through simulations and a real-world application to Seagate Technology's wafer manufacturing data, providing valuable insights for process optimization and real-time root cause tracing.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

In the age of information overload and massive data, the ability to generate concise and accurate summaries of lengthy documents or understand causal relationships within complex systems is vital. The primary goal of this thesis is to develop a new set of methods to tackle with these two problems.

The first part of the thesis is about *abstractive summarization*, which involves generating concise summaries by rephrasing and introducing novel words to capture the most salient information in the source text. The first project delves into multi-sentence abstractive summarization, focusing on better incorporation of topic information and reducing repetitions in generated text. In this study, I propose a hierarchical approach to incorporating topic information by leveraging both document-level and sentence-level topic distributions. Moreover, a sentence-level penalized attention mechanism is introduced to address the repetition problem, resulting in more coherent and informative summaries.

The second project investigates the potential of normalizing flows to improve abstractive summarization. While existing variational models address issues such as exposure bias and lack of text generation diversity, they often face challenges in capturing complex semantic structures and dealing with posterior collapse during training. In this work, I introduce FlowSUM, a normalizing flows-based variational encoder-decoder (VED) framework for Transformer-based summarization, along with a controlled alternate aggressive training strategy and a refined gate mechanism to overcome these challenges. FlowSUM demonstrates the effectiveness of normalizing flows in enriching the latent posterior distribution and improving summary quality, while also facilitating knowledge distillation. The findings of this project provide valuable insights on the operating characteristics of normalizing flows in summarization and imply the potential of normalizing flows in transferring knowledge from advanced large language models, shedding light on future directions in the field.

The second part of the thesis is about *causal discovery*, the problem of learning the causal relationships from observational data. The third project in the thesis centers on causal structure learning in the wafer manufacturing domain, specifically targeting the causal relations among sensors and abnormal events in a wafer assembly line. Understanding these causal relationships is essential for optimizing the manufacturing process, identifying root causes of failures, and providing real-time error corrections. By proposing a hierarchical ensemble approach that leverages temporal and domain constraints, we simultaneously address the challenges of high-dimensional data, mixed data types, imbalanced data, irregular missing patterns, and incorporating domain and temporal knowledge. The effectiveness of this approach is demonstrated through simulations and an application to wafer manufacturing data from Seagate Technology, receiving positive feedback from engineers and technicians.

By investigating these three projects, this thesis aims to shed light on the significant roles that topic awareness, normalizing flows, and hierarchical ensemble techniques can play in enhancing text summarization and causal discovery in manufacturing respectively. This research work stands to act as a catalyst for future inquiries in these domains, paving the way for more efficient and effective approaches to comprehend and distill extensive volumes of text, as well as uncovering causal relationships within complex systems.

The structure of this thesis is bifurcated into two main sections: part I delves into the realm of *abstractive summarization*, whereas part II pertains to *causal discovery*. These two components have been designed as self-contained units, enabling independent reading. In addition to these, a chapter detailing the software tools developed during my graduate career is included, as well as appendices that provide further context to the three projects.

Part I

Abstractive Summarization

Chapter 2

Text Summarization with Topic Awareness and Penalized Attention

2.1 Introduction

Automatic summarization focuses on capturing the most salient information in the source text and producing a condensed shorter version of the text. Based on the approach, there are two categories: extractive summarization (Cheng and Lapata, 2016; Nallapati et al., 2017) identifies and concatenates parts of the input document, while abstractive summarization (See et al., 2017b; Paulus et al., 2018; Wang et al., 2018) focuses on generating a summary by rephrasing or introducing novel words and is in general more challenging. Based on the output, there are tasks targeting at headline summarization (Rush et al., 2015), single-sentence summarization (Narayan et al., 2018b), and multi-sentence summarization (See et al., 2017b), the last of which we believe is more widely applicable. In this chapter, we focus on multi-sentence abstractive summarization, for which a lot of progress has been made with neural-based sequence-to-sequence models (Nallapati et al., 2016; See et al., 2017b) and pre-trained lanugage models (Liu and Lapata, 2019a; Zhang et al., 2020a; Rothe et al., 2020a; Raffel et al., 2020). To generate summaries with higher qualities, we focus on two major aspects: how to better incorporate topic information and how to reduce repetitions in the generated text.

Leveraging topic information in summarization has received growing research interest in recent years. One major motivation is that sequence-to-sequence models usually focus on local and sequential information, while topic representations can capture the global semantic[1] information of the document and the word co-occurrence statistics at the corpus-level (Ailem et al., 2019). Such information provides a richer global context for language generation and hence can be a valuable booster for text summarization models. Many of existing work used the output from external topic models such as LDA (Blei et al., 2003). However, the marriage between a neural model and an externally trained topic model is doubtful, as shown empirically in the experiments in Dieng et al. (2017). Also, topic models usually exert assumptions that the source text does not necessarily satisfy. For example, LDA assumes that the topic proportions follow a Dirichlet distribution and that the topic assignment follows a multinomial distribution, which may not be applicable for articles in diverse domains. These assumptions restrict the flexibility of topic modeling and hence potentially impact the effect of topic incorporation in the summarization models. Therefore, jointly performing summarization and topic inference might help better capture the topic information that is the most important for summarization.

At the same time, we notice the repetition problem that many RNN-based models can easily get into. As illustrated in Table 2.1, the repetition in the generated text greatly affects the summary quality. Three major approaches have been proposed to solve this problem in the existing work: manipulating the attention mechanism (Sankaran et al., 2016; Tu et al., 2016), modifying the objective function (Welleck et al., 2019), and proposing new decoding strategies (Fan et al., 2018; Holtzman et al., 2020). By checking the attention heatmaps in the baseline model, we identify some

[1]Topic models model on the joint distribution of the words in a document, while seq2seq models focus on local conditional distribution. That's what we mean global semantics and local semantics.

non-negligible repetition patterns which are closely related to the repeated text in the generated summary, as shown in Figure 2.1a. In addition, we observe that repetitions usually occur between sentences but rather inside a sentence. With this, we find the first approach more intuitive and propose a sentence-level penalized attention mechanism to refrain the decoder from attending to the already attended regions in the previous sentences.

The main contribution of this chapter are four-fold. First, we incorporate the topic information in a hierarchical manner where we fuse the document-level topic representation into the word embeddings and capture finer-grained topic representations via a topic attention module. Second, we alleviate the repetition problem in multi-sentence summaries with a penalized attention mechanism, enforced in both the encoder attention module and the topic attention module. Third, we validate the proposed model with extensive experiments on three multi-sentence benchmark datasets, CNN/Daily Mail (Nallapati et al., 2016), Multi-News (Fabbri et al., 2019b) and NEWSROOM (Grusky et al., 2018), exhibiting great improvement over the ROUGE scores and the repetition measures. Fourth, we analyze the attention heatmaps and demonstrate when adding attention penalties can be helpful and when cannot.

The rest of the chapter goes as follows. Section 2.2 describes the architecture of the proposed model in detail; Section 2.3 briefly summarizes the related work; Section 2.4 gives the implementation settings and shows the experiment results; Section 2.5 concludes this chapter with some discussion.

Original Text (truncated): (CNN)If you've been following the news lately, there are certain things you doubtless know about Mohammad Javad Zarif. He is, of course, the Iranian foreign minister. He has been U.S. Secretary of State John Kerry's opposite number in securing a breakthrough in nuclear discussions that could lead to an end to sanctions against Iran – if the details can be worked out in the coming weeks. And he received a hero's welcome as he arrived in Iran on a sunny Friday morning. "Long live Zarif," crowds chanted as his car rolled slowly down the packed street. You may well have read that he is "polished" and, unusually for one burdened with such weighty issues, "jovial."

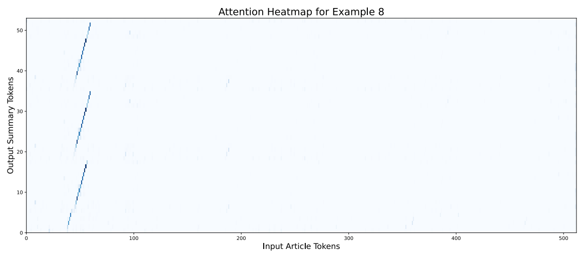
Ground Truth Summary: Mohammad Javad Zarif has spent more time with John Kerry than any other foreign minister. He once participated in a takeover of the Iranian Consulate in San Francisco. The Iranian foreign minister tweets in English.

Seq2seq + Encoder Attention: the iranian foreign minister has been u.s. secretary of state john kerry's opposite number. he has been u.s. secretary of state john kerry's opposite number. he has been u.s. secretary of state john kerry's opposite number.

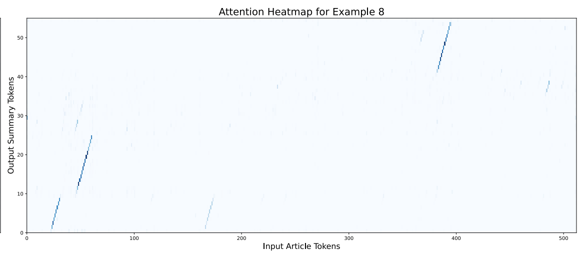
Seq2seq + Encoder Attention + Topic Attention: the iranian foreign minister has been u.s. secretary of state john kerry's opposite number. he has been u.s. secretary of state john kerry's opposite number. he has been u.s. secretary of state john kerry's opposite number.

Seq2seq + Penalized Encoder Attention + Penalized Topic Attention: mohammad javad zarif has been u.s. secretary of state john kerry's opposite number. he has been u.s. secretary of the country in 1977. he was nominated to be foreign minister by ahmadinejad's successor.

Table 2.1: Comparison of 3 models in handling repetition for Example 8 in the CNN/Daily Mail test set. The repeated text is highlighted in green.



(a) Baseline Attention Heatmap for Example 8



(b) PA-TAM Attention Heatmap for Example 8

Figure 2.1: Encoder Attention Maps for the CNN/Daily Mail test dataset. The x-axis represents the input article tokens and the y-axis represents the output summary tokens. The x-ticks are the same for the two subplots, but the y-ticks can be different, since different models generate summaries of different lengths. The heatmap is based on the `plt.cm.Blues` colormap[2], where the value of lightness monotonically increases with the attention weight value.

2.2 Model

Our proposed model consists of three key components: document-level topic embedding, sentence-level encoder-decoder penalized attention, and sentence-level topic-decoder penalized attention, as described in Figure 2.2. Both the topic embedding component and the topic-decoder attention mechanism are closely related to a pool of topic latent representation vectors. The topic embedding component takes in the average embedding of the entire input text and learns a topic proportions vector across the topic pool, upon which a document-level topic context vector is constructed. The topic-decoder attention component runs at each decoding step, producing a topic context for each word. A highlight of our model is the penalty mechanism applied to both the encoder and the topic attention component, so as to reduce the repetition appearing across different sentences. Note that both penalties are applied after the generation of a full sentence, instead of at the word-level.

[2] `https://matplotlib.org/3.5.0/tutorials/colors/colormaps.html#sequential`

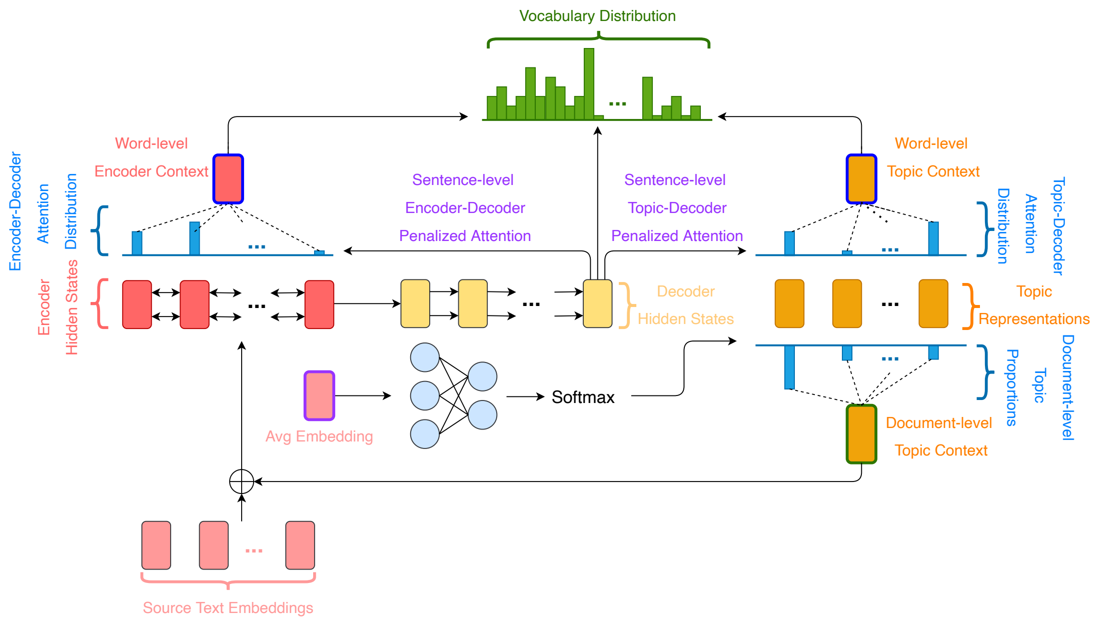


Figure 2.2: PA-TAM Model Architecture.

Throughout this section, let e be the embedding size, V be the vocabulary size, m be the length of the input source text, and n be the length of the target summary text. Let $\mathbf{E} = (E_1, E_2, \cdots, E_V) \in \mathbb{R}^{e \times V}$ be the embedding matrix for the whole vocabulary[3], $\boldsymbol{x} = (x_1, x_2, \cdots, x_m) \in \mathbb{R}^{e \times m}$ be the word embeddings of the input source text, $\boldsymbol{y} = (y_1, y_2, \cdots, y_n) \in \mathbb{R}^{e \times n}$ be the word embeddings of the target summary text, and $\boldsymbol{b} = (b_1, b_2, \cdots, b_V)^T \in \mathbb{R}^{V \times 1}$ be the Bag-of-Words[4] vector of the input source text. Note that if we don't truncate the input source text, then we have $b^T \mathbb{1} = m$, and if we do truncate the input source text due to the encoder's limitation, then we have $b^T \mathbb{1} > m$. Also, let t_ℓ denote the starting word index of the ℓth sentence and $(y_{t_\ell}, y_{t_\ell + 1}, \cdots, y_{t_{\ell+1} - 1})$ denote the ℓth sentence in the summary, where $\ell \in \{1, 2, \cdots, L\}$ and L is the total number of sentences in the summary. Additionally, we let $\mathbf{T} = (T_1, T_2, \cdots, T_K)$ be a pool of topic latent representation vectors, where

[3] A set of words under consideration. If outside this set, then identified as UNK (unknown).
[4] The frequency of each word in the vocabulary.

$T_k \in \mathbb{R}^{e\times 1}$ and K is the number of topics.

2.2.1 Document-level Topic Embedding

We first calculate the average embedding of the untruncated input source text as

$$E_{\text{avg}} = \frac{\sum_{v=1}^{V} b_v E_v}{\sum_{v=1}^{V} b_v} \in \mathbb{R}^{e\times 1}.$$

Then we pass the average embedding vector into a neural network and a softmax layer to obtain a document-level topic proportions vector:

$$\theta = \text{softmax}\left(\text{NN}\left(E_{\text{avg}}\right)\right) \in \mathbb{R}^{K\times 1}.$$

Next, we calculate the document-level topic context vector using the topic proportions and the topic latent vectors:

$$T_{\text{doc}} = \sum_{k=1}^{K} \theta_k T_k \in \mathbb{R}^{e\times 1}.$$

Finally, we integrate the source text embedding vectors $\boldsymbol{x}$ with T_{doc} and obtain topic-enhanced embedding vectors as

$$\boldsymbol{x}' = (x'_1, x'_2, \cdots, x'_m) = (x_1 + T_{\text{doc}}, x_2 + T_{\text{doc}}, \cdots, x_m + T_{\text{doc}}) \in \mathbb{R}^{e\times m}.$$

2.2.2 Sequence-to-sequence Framework

We use a bidirectional LSTM as the encoder and take the topic-enhanced embedding vectors $\boldsymbol{x}'$ as the input. We represent the encoder hidden states and cell states as follows.

$$h_i^{enc} = [\overleftarrow{h_i^{enc}}; \overrightarrow{h_i^{enc}}] \in \mathbb{R}^{2h\times 1}, 1 \le i \le m;$$

$$c_i^{enc} = [\overleftarrow{c_i^{enc}}; \overrightarrow{c_i^{enc}}] \in \mathbb{R}^{2h\times 1}, 1 \le i \le m.$$

We initialize the decoder's first hidden state and cell state with a linear projection of the encoder's final hidden and cell state.

$$h_0^{dec} = W_h[\overleftarrow{h_1^{enc}}; \overrightarrow{h_m^{enc}}] \in \mathbb{R}^{h\times 1},\ \text{where } W_h \in \mathbb{R}^{h\times 2h};$$

$$c_0^{dec} = W_c[\overleftarrow{c_1^{enc}}; \overrightarrow{c_m^{enc}}] \in \mathbb{R}^{h\times 1},\ \text{where } W_c \in \mathbb{R}^{h\times 2h}.$$

Next, we feed in the target summary. At the tth step, for the tth word w_t, get its embedding vector $y_t \in \mathbb{R}^{e\times 1}$ and concatenate it with the combined-output vector $o_{t-1} \in \mathbb{R}^{h\times 1}$ (defined later) from the previous step (initialize o_0 with a zero-vector) and get $\overline{y}_t = [y_t; o_{t-1}] \in \mathbb{R}^{(h+e)\times 1}$. Finally, feed $\overline{y}_t$ as an input to the LSTM decoder and make a step forward.

$$h_t^{dec}, c_t^{dec} = \text{Decoder}(\overline{y}_t, h_{t-1}^{dec}, c_{t-1}^{dec}) \in \mathbb{R}^{h\times 1}.$$

2.2.3 Encoder-Decoder Attention Penalized at the Sentence Level

We use h_t^{dec} to compute the attention over $h_1^{enc}, h_2^{enc}, \cdots, h_m^{enc}$ and calculate the word-level encoder context vector h_t^*. There are many types of attention alignment choices, such as the basic dot-product attention, the multiplicative attention, and the additive attention. In this chapter, we consider the Bahdanau attention mechanism as in Bahdanau et al. (2015). Let Alignment$(\cdot)$ denote the attention weight calculation. The penalized encoder-decoder attention mechanism goes as follows. We initialize the penalty for the first sentence as zeros $\gamma_1 = \mathbf{0} \in \mathbb{R}^{m\times 1}$. When $t_\ell \le t \le t_{\ell+1} - 1$, namely, when the tth word is in the ℓth sentence, we first calculate the alignment e_i^t between the encoder hidden state h_i^{enc} and the decoder hidden state h_t^{dec}, then we apply the penalty γ_ℓ to the attention alignment scores, followed by a softmax transformation, to get the penalized attention weights a^t, and finally, we obtain the encoder context vector h_t^* as a weighted encoder hidden state.

$$e_i^t = \text{Alignment}(h_t^{dec}, h_i^{enc}) = W_{combined\_enc\_att}\tanh(W_{dec\_enc\_att}h_t^{dec}+W_{enc\_att}h_i^{enc}),$$

where $1 \le i \le m, e^t \in \mathbb{R}^{m\times 1}, W_{combined\_enc\_att} \in \mathbb{R}^{1\times h}, W_{dec\_enc\_att} \in \mathbb{R}^{h\times h}, W_{enc\_att} \in \mathbb{R}^{h\times 2h}$;

$$a^t = \text{softmax}\left(e^t + \gamma_\ell\right) \in \mathbb{R}^{m\times 1};$$

$$h_t^* = \sum_i a_i^t h_i^{enc} \in \mathbb{R}^{2h\times 1}.$$

When $t = t_{\ell+1} - 1$, namely, when the word is at the end of the ℓth sentence, we use max pooling to aggregate the attention weights in this sentence and calculate the attention penalty for the next sentence.

$$(a_{\max}^{(\ell)})_i = \max_{t_\ell \le t \le t_{\ell+1}-1} a_i^t, \quad 1 \le i \le m;$$

$$\gamma_{\ell+1} = \gamma_\ell + f(a_{\max}^{(\ell)}) \in \mathbb{R}^{m\times 1},$$

where $f(x) = 1 - \frac{1}{1-x}$ is a transformation function.

There are two main reasons for using max pooling as the aggregation method. First, if we use sum pooling, then for long sentences, even insignificant attention may turn out to be significant cumulatively. In contrast, max pooling allows us to focus on the most salient part and will not be affected by the sentence length. Second, we observe that the words in the same sentence usually do not attend to the same token in the source text, and therefore, we will not lose much as compared to sum pooling for short sentences.

Regarding the transformation function f, it can be any function that satisfies the following requirements: (1) $f : [0, 1) \to (-\inf, 0]$; (2) $f(0) = 0$; (3) $\lim_{x\to 1} f(x) = -\inf$; (4) f is a decreasing function in $[0, 1)$. Intuitively, when the previous attention weight is 0, we don't penalize the token; when the previous attention weight is close to 1, the maximum value, we penalize the token such that it would not occur any longer; and we exert a harsher penalty if the token has been attended more often in the previous sentences.

2.2.4 Topic-Decoder Attention Penalized at the Sentence Level

To capture the topic information at a finer granularity, we propose a topic attention module. Unlike the encoder attention where the encoder hidden states are different across input samples, the topic attention module utilizes a pool of latent vectors which is shared by all data samples. The original attention mechanism in neural machine translation (Bahdanau et al., 2015) was designed to explicitly establish a soft alignment between the decoding step and the input text. It can be seen as a dynamic feature selector which selects different subset of input text features for word generation. Following the same logic, by applying attention to the topic latent vectors, we allow the decoder to dynamically select topic features for word generation.

Similar to the encoder-decoder attention, we use h_t^{dec} to compute the attention over the topic latent representations $T_1, T_2, \cdots, T_K$ and we add a penalty mechanism at the sentence-level. We initialize the penalty for the first sentence as $\nu_1 = \mathbf{0} \in \mathbb{R}^{K\times 1}$. When the tth word is in the ℓth sentence, namely, $t_\ell \leq t \leq t_{\ell+1} - 1$, we calculate the alignment ξ_k^t between the topic latent vector T_k and the decoder hidden state h_t^{dec}, then we get the penalized attention weight η^t, and finally, we obtain the topic context vector T_t^* as a weighted topic latent representation.

$$\xi_k^t = \text{Alignment}(h_t^{dec}, T_k) = W_{combined\_topic\_att} \tanh(W_{dec\_topic\_att} h_t^{dec} + W_{topic\_att} T_k),$$

where $1 \leq k \leq K, \xi^t \in \mathbb{R}^{K\times 1}$, $W_{combined\_topic\_att} \in \mathbb{R}^{1\times h}$, $W_{dec\_enc\_att} \in \mathbb{R}^{h\times h}$, $W_{topic\_att} \in \mathbb{R}^{h\times e}$;

$$\eta^t = \text{softmax}(\xi^t + \nu_\ell) \in \mathbb{R}^{K\times 1};$$

$$T_t^* = \sum_k \eta_k^t T_k \in \mathbb{R}^{e\times 1}.$$

When $t = t_{\ell+1} - 1$, we use sum pooling to aggregate the attention weights in this sentence, upon which we calculate the attention penalty for the next sentence. Note that we replace max pooling with sum pooling here, and the main reason is that in topic attention, the words in the same sentence tend to attend to similar topics, which makes sum pooling a more sensible choice. Also, the choice of the transformation function f can be different from the one in the encoder-decoder attention penalization, but we use the same one in this chapter.

$$\left(\eta_{\text{cum}}^{(\ell)}\right)_k = \max\left(1 - 10^{-6}, \sum_{t=t_\ell}^{t_{\ell+1}-1} \eta_k^t\right), \quad 1 \leq k \leq K;$$

$$\nu_{\ell+1} = \nu_\ell + f(\eta_{\text{cum}}^{(\ell)}) \in \mathbb{R}^{K\times 1}.$$

Vocabulary Distribution We now concatenate the decoder hidden state h_t^{dec} with the encoder context vector h_t^* and the topic context vector T_t^*. Then, we pass it through a linear layer, a tanh layer and a dropout layer to obtain the combined-output vector o_t, which is then used to generate the vocabulary distribution.

$$u_t = [h_t^{dec}; h_t^*; T_t^*] \in \mathbb{R}^{(3h+e)\times 1};$$

$$v_t = W_u u_t \in \mathbb{R}^{h\times 1}, \text{ where } W_u \in \mathbb{R}^{h\times(3h+e)};$$

$$o_t = \text{dropout}(\tanh(v_t)) \in \mathbb{R}^{h\times 1};$$

$$P_t = \text{softmax}(W_{\text{vocab}} o_t) \in \mathbb{R}^{V\times 1}, \text{ where } W_{\text{vocab}} \in \mathbb{R}^{V\times h}.$$

Loss We use cross-entropy as the loss function. Let g_t be the one-hot vector of the target word w_t at timestep t and Θ represent all parameters of the model. Then the loss function can be calculated as below.

$$J_t(\Theta) = CrossEntropy(g_t, P_t) = -\log(P_t(w_t));$$

$$J = \frac{1}{n}\sum_{t=1}^{n} J_t(\Theta).$$

2.3 Related Work

Although pre-trained language models (Liu and Lapata, 2019a; Zhang et al., 2020a; Rothe et al., 2020a; Raffel et al., 2020) have become the default choices in many applications, the traditional neural attention sequence-to-sequence model framework (Rush et al., 2015; Chopra et al., 2016; Nallapati et al., 2016) still remains popular. One star model that is built on top of it is the pointer-generator network (See et al., 2017b), which has been used as the baseline for model comparison in many papers due to its good performance (Narayan et al., 2018b; Ailem et al., 2019; Wang et al., 2020a; Fu et al., 2020). Additionally, the simplicity of this framework in implementation and attention visualization allows us to better examine the effect of the newly proposed mechanisms. Therefore, our model adopts this framework to demonstrate the effectiveness of our topic incorporation mechanism and penalized attention mechanism.

Incorporating topic information in text generation and summarization has gained growing popularity in recent years. Many of the existing work used external topic models such as LDA (Blei et al., 2003). Narayan et al. (2018b) passed the distributions obtained from LDA as an additional input and concatenated the word embeddings with the point-wise multiplication of the topic distribution of the document and the topic distributions of the words to obtain topic sensitive embeddings of the input. Wang et al. (2018) passed the word embeddings and the topic embeddings obtained by LDA to two convolutional blocks separately and incorporated the topic information by a joint attention and biased probability generation mechanism. Ailem et al. (2019) leveraged topic information by transforming the inner product of the topic-word parameters and the topic vector, both given by LDA, into a new mixture component in the generative probability for decoding. However, the combination between a neural model and an externally trained topic model does not guarantee a good summarization model, as shown in the experiments in Dieng et al. (2017). Also, the assumptions innate to those topic models are likely to be unsatisfied. Realizing these potential issues of external topic models, Fu et al. (2020) proposed to accomplish topic inference and summarization in an end-to-end manner via a variational encoder-decoder framework with both the paragraph-level and document-level latent topics. They chose the Gaussian distribution as the prior for the latent topic vector and used variational inference to model the topics. In modeling the paragraph-level latent topics, they first decomposed the input text into a fixed number of paragraphs and then passed the bag-of-words of these paragraphs to obtain paragraph-level topics, upon which they then applied topic attention in decoding. However, their paragraph division introduced an additional hyper-parameter to tune and was just based on word count, which might lead to meaningless divisions in that the content inside a segment may cover totally different topics. Also, the variational encoder-decoder framework can easily into the KL explosion and KL vanishing problem (Fu et al., 2019), showing high training instability. Our model uses similar ideas as Fu et al. (2020) for the document-level topic embeddings, but we use a different topic attention mechanism to capture the lower-level topic information, where we allow the word-level latent topics share the same topic representation pool as the document-level topics and we get rid of the variational networks to avoid the high training instability.

The repetition problem of text generation has also been extensively studied to improve the text quality (Tu et al., 2016; Sankaran et al., 2016; Mi et al., 2016). Sankaran et al. (2016) proposed a temporal attention model in neural machine translation, where they directly divided the attention distribution by the sum of the previous attention weights, which had been shown to greatly improve the translation quality. See et al. (2017b) proposed a coverage mechanism in text summarization based on the coverage model by Tu et al. (2016) and showed great improvements in the ROUGE scores. This coverage mechanism included a coverage vector in the attention calculation and introduced a coverage loss in the objective function. However, as their experiments showed, the modified attention mechanism had to be coupled with the coverage loss to be effective and a two-phase training procedure was needed to avoid overall performance reduction caused by the additional coverage loss. Holtzman et al. (2020) pointed out the standard likelihood training and the maximization-based decoding can lead to text degeneration, where the output text would be incoherent or get stuck in repetitive loops, and proposed nucleus sampling as the decoding strategy to overcome the issues. Welleck et al. (2019) proposed an unlikelihood training technique to fix the standard likelihood training issue. They treated previous generated tokens as negative candidates and defined an unlikelihood loss based on them, after which they then added the unlikelihood loss to the standard likelihood as the training objective function. Although this technique did help reduce the repetition measures, it increased the perplexity at the same time due to the additional loss term, which may raise concerns in scenarios where perplexity is important. Nair and Singh (2021) adopted the unlikelihood training technique and extended the coverage and temporal attention mechanisms (Sankaran et al., 2016) to the token level and empirically showed that these techniques jointly helped reduce repetitions and increase ROUGE scores on the CNN/Daily Mail dataset. Liu et al. (2021) tackled the repetition problem by jointly improving the attention mechanism and the decoding procedure. They identified repetition patterns in the attention heatmaps in the convolutional seq2seq models and then proposed a segment-wise attention filter mechanism and a sentence-level backtracking decoder. They showed that their proposal outperformed the other repetition reduction techniques (Nallapati et al., 2016; See et al., 2017b; Paulus et al., 2018; Gehrmann et al., 2018; Celikyilmaz et al., 2018) on the CNN/Daily Mail dataset. The heatmaps revealed in their paper greatly motivate our penalized attention mechanism. Different from their work, we observe that repetitions rarely occur inside a sentence but more often between sentences, and we argue that different sentences should cover different topics and the words in the same sentence should belong to similar topics. Therefore, instead of using segment-wise attention weights to revise the attention distribution as in Liu et al. (2021), we aggregate and modify the attention weights at the sentence level, for both the attention to the encoder hidden states and to the topic latent vectors.

2.4 Experiments

2.4.1 Setup

Datasets

We use three benchmark datasets for the experiments: CNN/Daily Mail (Hermann et al., 2015), Multi-News (Fabbri et al., 2019b) and NEWSROOM (Grusky et al., 2018). These three datasets are all in the news area and have been widely used as the benchmark for summarization tasks.

CNN/Daily Mail consists of 312,085 online news articles, where each article (781 tokens on average) is paired with a multi-sentence summary (3.75 sentences on average). As in See et al. (2017b), we use the non-anonymized version of the data, which is split into 287,226, 13,368, and 11,490 pairs for training, validation and testing respectively.

Multi-News contains 56k pairs of news articles and summaries. This is a multi-document dataset, and on average, there are 2103 tokens in each article and 9.97 sentences in each summary. We use the same truncated version as in Fabbri et al. (2019b), where for each example with S source documents, the truncated article is the concatenation of the first $500/S$ tokens from each source document. The whole dataset is split into 44,972, 5,622, and 5,622 for training, validation, and testing respectively.

NEWSROOM collects 1.3M pairs of articles and summaries written by authors and editors from the newsrooms of 38 major news publications from 1998 to 2017. On average, there are 658.6 words in each article and 26.7 words in each summary. And there are 995,041, 108,837, 108,862 examples in the training, validation, and testing dataset respectively.

Models in comparison

The baseline model is an LSTM-based sequence-to-sequence model with the plain attention mechanism. For our proposal, we consider three variants: penalized attention baseline model (PA-Baseline), topic-aware model (TAM) and penalized attention topic-aware model (PA-TAM). PA-Baseline adds penalties to the encoder attention mechanism in the baseline model. TAM adds the topic attention and the topic embedding component on top of the baseline model, and PA-TAM adds penalties to both the encoder attention and the topic attention in the TAM.

Evaluation metrics

We use perplexity, the standard ROUGE scores(Lin, 2004a) and three repetition measures to evaluate the models. Perplexity is the standard evaluation metric for language models, as defined in Equation (2.1). Note that it equals to the exponential of the cross-entropy loss $J(\Theta)$ and a smaller perplexity suggests a better model. The ROUGE metrics are commonly used in machine translation and text summarization. They compare the automatically generated summary or translation against the reference(s). In this chapter, we consider ROUGE-1, ROUGE-2 and ROUGE-L, which respectively refers to the overlap of unigrams, bigrams, and the longest common sequence between the generated summary and the reference summary. We obtain the ROUGE scores using the `compare_mt` package[5] and we report the F_1 scores for these three metrics.

Regarding the repetition measures, we use rep-w, rep-n and rep-r as in Fu et al. (2021). As a token-level metric, rep-w measures the proportion of words that occur in the previous w tokens, where w is a bandwidth hyper-parameter, which is set as 16 in this chapter. At the segment level, we use rep-n to measure the duplication of n-grams in the text and we set $n = 3$ in the chapter. The rep-r metric is proposed by Fu et al. (2021) to avoid the dependence on hyper-parameters and it measures the proportion of words that appear in a repeated segment. Let s represent the text, $|s|$ be the number of tokens in the text and s_i be the ith token. These three measures can then be evaluated by Equation (2.3), (2.4) and (2.5) respectively.

$$\begin{aligned} \text{perplexity} &= \prod_{t=1}^{n} \left(\frac{1}{P_{\text{LM}}\left(w_t \mid w_{t-1}, \ldots, w_1\right)} \right)^{1/n} \quad (2.1) \\ &= \prod_{t=1}^{n} \left(\frac{1}{P_t(w_t)} \right)^{1/n} = \exp\left(\frac{1}{n} \sum_{t=1}^{n} - \log P_t(w_t) \right) = \exp(J(\Theta)) \quad (2.2) \end{aligned}$$

$$\text{rep-w} = \frac{1}{|s|} \sum_{t=2}^{|s|} \mathbb{1}\left[s_t \in s_{\max(t-w,1):t-1} \right] \quad (2.3)$$

[5] https://github.com/neulab/compare-mt

$$\text{rep-n} = 1.0 - \frac{|\text{unique n-grams}|}{|\text{n-grams}|} = 1 - \frac{|\{\tilde{s} \mid \exists i \in [1, |s| - n + 1], \text{s.t. } \tilde{s} = s_{i:i+n-1}\}|}{|s| - n + 1} \tag{2.4}$$

$$\text{rep-r} = \frac{|\{i \mid (\exists j \neq i, s_i = s_j \text{ and } s_{i+1} = s_{j+1}) \text{ or } (\exists k \neq i, s_i = s_k \text{ and } s_{i-1} = s_{k-1})\}|}{|s|} \tag{2.5}$$

Implementation Details

For all models, the hidden size is 300, batch size is 16, beam search size is 5, embedding size is 768. We use the word embeddings given by the RoBERTa Tokenizer (Liu et al., 2019), which has a vocabulary of 50,265 tokens. We use different number of topics K for different datasets: $K = 200$ for the CNN/Daily Mail dataset, and $K = 150$ for the Multi-News and the NEWSROOM dataset. The neural network in the document-level topic modeling component is a three-layer feed-forward neural net: $\text{FF}(768, 800), \text{LeakyReLU}, \text{FF}(800, 800), \text{LeakyReLU}$, Dropout(0.2), $\text{FF}(800, K)$.

We train the models using the Adam optimizer (Kingma and Ba, 2015) with an L_2 penalty of value 1.2e-6. The initial learning rate differs across different datasets and the learning rate decreases exponentially till it reaches a minimum threshold 0.002. We use the gradient clipping technique with a maximum gradient norm of 1.0 or 2.0, depending on the models and the datasets. See the full details in Appendix B.1.

2.4.2 Quantitative Results

We evaluate our proposed model by comparisons with the baseline model. The results are given in Table 2.2 - Table 2.4. Table 2.2 shows the results on the CNN/Daily Mail dataset. It presents that TAM leads to +0.94, +0.54, +0.78 improvements over the baseline model in terms of R-1, R-2 and R-L as well as minor reductions in terms of the repetition measures, implying that the inclusion of topic information does help capture additional information and improve the summary quality. PA-Baseline achieves major improvements over the baseline model, with +2.45, +1.48 increments in terms of R-1 and R-2 respectively and -0.0947, -0.1532, -0.2705 reductions in terms of rep-w, rep-3 and rep-r respectively. The R-L score, however, suffers a -4.77 reduction, suggesting that adding penalties to the encoder attention only may not be sufficient. PA-TAM combines the best of both the TAM and the PA-Baseline model and gains further improvements over TAM with +2.82, +1.47, +2.82 improvements with regard to R-1, R-2 and R-L as well as major reductions in terms of perplexity and all three repetition measures. Its good performance indicates that by including the topic information and by exerting the attention penalties, PA-TAM can produce summaries with higher qualities. Table 2.3 shows similar results on the Multi-News dataset, except that the PA-Baseline model has much higher perplexity and rep-w measure than the baseline model. Still, PA-TAM is the best performing model.

Table 2.4, however, shows slightly different patterns on the NEWSROOM dataset. First, the improvements of the proposed models over the baseline model are very small. Second, TAM is better than PA-Baseline in terms of perplexity and ROUGE scores. Third, though PA-TAM is still the best overall, TAM has the smallest perplexity. After checking the repetition measures of the oracle summaries, we find that the NEWSROOM data is much less repetitive than the CNN/Daily Mail and Multi-News data and that the repetition gap between the baseline and the oracle is also smaller for the NEWSROOM data, which explains why adding penalties does not boost the

Method	Perplexity	ROUGE			Repetition		
		1	2	L	rep-w	rep-3	rep-r
Baseline	20.9504	32.93	13.69	30.06	0.2511	0.2296	0.4866
TAM	20.4605	33.87	14.23	30.84	0.2235	0.2153	0.4515
PA-Baseline	20.9927	35.38	15.17	25.29	0.1564	0.0764	0.2161
PA-TAM	**18.8701**	**36.69**	**15.70**	**33.66**	**0.1450**	**0.0687**	**0.2022**
Oracle					0.1124	0.0017	0.0436

Table 2.2: Perplexity and ROUGE F_1 scores on the CNN/Daily Mail test set.

Method	Perplexity	ROUGE			Repetition		
		1	2	L	rep-w	rep-3	rep-r
Baseline	35.1418	31.83	10.03	18.76	0.3091	0.4486	0.7289
TAM	36.1070	32.54	10.27	19.13	0.3087	0.4675	0.7495
PA-Baseline	43.5885	33.93	11.46	18.67	0.3608	0.2984	0.5747
PA-TAM	**34.6546**	**37.03**	**13.07**	**20.12**	**0.2688**	**0.2007**	**0.4569**
Oracle					0.1393	0.0127	0.2113

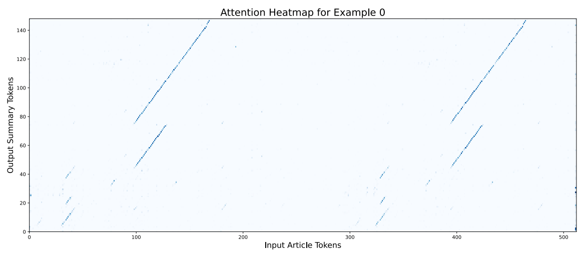
Table 2.3: Perplexity and ROUGE F_1 scores on the Multi-News test set.

model performance in this case.

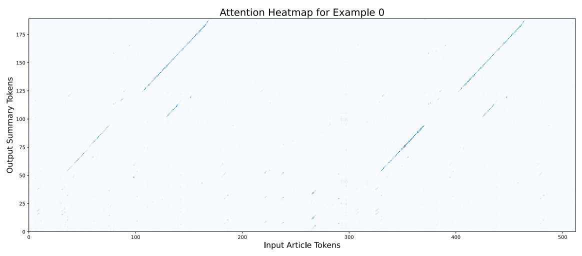
The heatmaps of the encoder attention weights also support our findings. As shown in Figure 2.1 and Figure 2.3, on the CNN/Daily Mail and the Multi-News dataset, the baseline model attends to similar regions repetitively, while the PA-TAM model removes such repetition in the encoder attention and hence increases the summary quality. In contrast, the repetitive attention problem rarely occurs in the baseline model on the NEWSROOM data, further explaining why the addition of penalty does not help much. Additionally, as shown in Figure 2.4, when there are repetitions in the baseline model, the attention penalty helps reduce the repetition, and when there are no repetitions, the addition of penalties shifts the attention region, which might have negative effects on the summary generation.

Method	Perplexity	ROUGE			Repetition		
		1	2	L	rep-w	rep-3	rep-r
Baseline	41.2379	37.94	26.39	34.18	0.0837	0.0089	0.0397
TAM	**38.1434**	38.23	26.58	34.44	0.0821	0.0089	0.0379
PA-Baseline	39.6249	38.20	26.58	34.38	0.0830	0.0076	0.0357
PA-TAM	38.3615	**38.34**	**26.66**	**34.53**	**0.0816**	**0.0066**	**0.0347**
Oracle					0.0572	0.0010	0.0142

Table 2.4: Perplexity and ROUGE F_1 scores on the NEWSROOM test set.

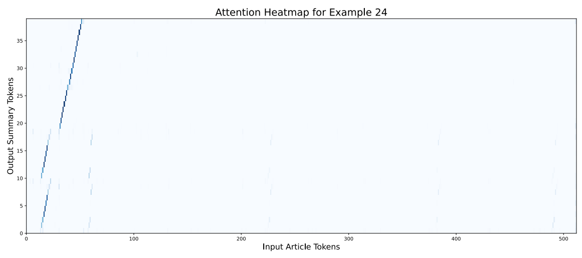


(a) Baseline Attention Map for Example 0

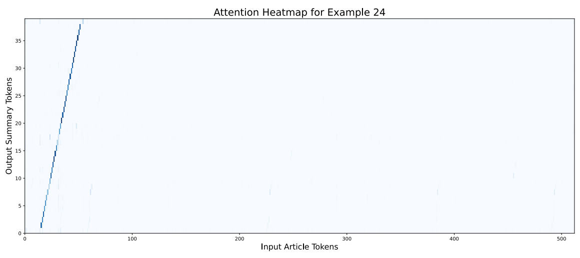


(b) PA-TAM Attention Map for Example 0

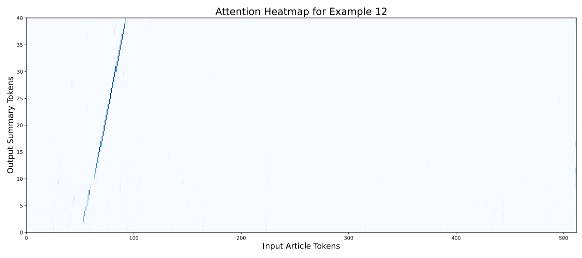
Figure 2.3: Encoder Attention Maps for the Multi-News test dataset. The x-axis represents the input article tokens and the y-axis represents the output summary tokens. The x-ticks are the same for the two subplots, but the y-ticks can be different, since different models generate summaries of different lengths. The heatmap is based on the `plt.cm.Blues` colormap, where the value of lightness monotonically increases with the attention weight value.



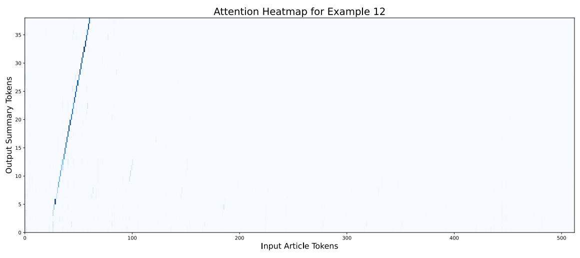
(a) Baseline Attention Map for Example 24



(b) PA-TAM Attention Map for Example 24



(c) Baseline Attention Map for Example 12



(d) PA-TAM Attention Map for Example 12

Figure 2.4: Encoder Attention Maps for the NEWSROOM test dataset. The x-axis represents the input article tokens and the y-axis represents the output summary tokens. The x-ticks are the same for the two subplots, but the y-ticks can be different, since different models generate summaries of different lengths. The heatmap is based on the `plt.cm.Blues` colormap, where the value of lightness monotonically increases with the attention weight value. Figure 2.4a and Figure 2.4b represent a scenario where there are repetitions in the baseline model and adding attention penalties helps avoid the repetition. Figure 2.4c and Figure 2.4d represent a scenario where no repetitions occur in the baseline model and adding attention penalties shifts the attention region leftward.

Method	Perplexity	ROUGE		
		1	2	L
seq2seq + enc_attn (Baseline)	20.9504	32.93	13.69	30.06
Baseline + topic_embed (BTM)	20.9895	**34.01**	14.05	30.80
Baseline + topic_embed&attn (TAM)	**20.4605**	33.87	**14.23**	**30.84**

Table 2.5: Ablation study for the topic-aware mechanism on the CNN/Daily Mail dataset. The reported metrics are evaluated on the test set.

2.4.3 Ablation Study

We perform the ablation study on the CNN/Daily Mail dataset to evaluate the effect of each individual component in the proposed model PA-TAM. We first examine the effect of topic information inclusion. As shown in Table 2.5, adding topic information to the word embeddings increases the ROUGE scores but also slightly increases the perplexity. And adding both the topic embedding and the topic attention leads to improvements in terms of all four measures. Next, we examine the effect of attention penalties. Table 2.6 shows that adding penalties to the encoder attention on top of Baseline, BTM and TAM all helps increase the summary quality and that the penalties on the topic attention further improves the model in terms of all metrics. Therefore, every component in the proposed model is contributing and they jointly make a better summarizer overall.

Method	Perplexity	ROUGE		
		1	2	L
seq2seq + enc_attn (Baseline)	20.9504	32.93	13.69	30.06
Baseline + enc_penalty (PA-Baseline)	20.9927	35.38	15.17	25.29
Baseline + topic_embed (BTM)	20.9895	34.01	14.05	30.80
BTM + enc_penalty	20.5693	36.27	15.46	33.30
Baseline + topic_embed&attn (TAM)	20.4605	33.87	14.23	30.84
TAM + topic_penalty	19.1229	34.14	14.33	31.11
TAM + enc_penalty	20.0342	36.44	15.64	33.55
TAM + enc_penalty + topic_penalty (PA-TAM)	**18.8701**	**36.69**	**15.70**	**33.66**

Table 2.6: Ablation study for the penalized attention mechanism on the CNN/Daily Mail dataset. The reported metrics are evaluated on the test set.

2.5 Conclusions and Discussions

In this chapter, we propose a summarization model with topic awareness and attention penalization, abbreviated as PA-TAM. This model captures the topic information through a topic embedding module and a topic attention module and reduces the repetition in the generated summaries via a penalized attention mechanism. Our experiments demonstrate the effectiveness of the proposed model in improving summary qualities and explicate under what scenarios would adding attention penalties be helpful.

At the same time, there are a few drawbacks of the current work. First, though we claim the learned latent vectors to be topic related due to the usage of Bag-of-Words representations, we cannot provide a topic interpretation for these latent vectors. We have tried adding topic interpretations by including an additional loss term, derived from the neural-based embedded topic model (Dieng et al., 2020), but the output summary has much lower qualities in terms of perplexity and ROUGE scores, suggesting the topic-related loss term interferes with the cross-entropy loss. Second, we have only showed the superiority of the proposed model over the baseline model so far, but have not yet compared with the other existing mechanisms for topic incorporation and repetition reduction. In the future, more experiments need to be done to examine the effect of the proposed mechanisms. Finally, the model backbone is LSTM-based, while the current state-of-the-art models are mostly based on Transformers. It is worth exploring further the application of the two proposed mechanism to the Transformer-based models.

Chapter 3

Boosting Summarization with Normalizing Flows and Aggressive Training

3.1 Introduction

Abstractive summarization (See et al., 2017a; Paulus et al., 2018; Wang et al., 2018) aims to generate summaries by rephrasing or introducing novel words to capture the most salient information in the source text. Many abstractive summarization models (Liu and Lapata, 2019b; Zhang et al., 2020a; Rothe et al., 2020b; Raffel et al., 2020) are based on the Transformers architecture (Vaswani et al., 2017) and have consistently produced state-of-the-art summarization quality. However, issues such as exposure bias (Ranzato et al., 2016; Qi et al., 2020), lack of text generation diversity (Holtzman et al., 2020), and insufficient capturing of semantic information (Reimers and Gurevych, 2019; Wang et al., 2020b) remain.

Variational models have gained increasing research interest (Zhang et al., 2016; Su et al., 2018; Wang et al., 2019; Fu et al., 2020) as they address these issues by introducing uncertainty in predictions through learning a probability distribution over latent variables. A variational model enables diverse text generation (Du et al., 2022), smoother output spaces, and semantically meaningful latent codes (Wang et al., 2019) that guide the generation of coherent and informative summaries.

Nonetheless, existing variational models have not fully achieved the aforementioned desirable properties due to two main challenges. Firstly, the semantic information in the source text may possess a complex structure. However, since introducing latent variables complicates parameter estimation, many current models (Fu et al., 2020; Zheng et al., 2020) represent latent codes using a Gaussian distribution, which is insufficient for capturing the intricacies of the latent space and could potentially reduce model performance. To enrich latent distributions, researchers suggest replacing the highly restricted isotropic Gaussian with normalizing flows (Rezende and Mohamed, 2015). Normalizing flows can generate complex distributions while preserving density in an analytical form, and they have been integrated into variational autoencoder (VAE) (Kingma and Welling, 2014) and variational encoder-decoder (VED) (Serban et al., 2017; Zhou and Neubig, 2017) frameworks to better approximate the latent posterior. For example, this approach has been used in text generation by Wang et al. (2019), in neural machine translation by Setiawan et al. (2020), and in dialogue generation by Luo and Chien (2021). Despite this progress, the operating characteristics of normalizing flows on summarization tasks have yet to be investigated.

Secondly, as reported by previous studies (Bowman et al., 2016; Kingma et al., 2016; Chen et al., 2017), variational models tend to experience posterior collapse during training, which occurs when the KL term vanishes to zero, indicating that the model fails to learn meaningful latent codes. This problem becomes more severe when modeling discrete data with a strong auto-regressive decoder (He et al., 2019), which is the case for Transformer-based summarization models. To resolve this issue, several solutions have been proposed, such as employing a less auto-regressive decoder network (Yang et al., 2017; Semeniuta et al., 2017; Shen et al., 2018), modifying the training objective (Zhao et al., 2017; Tolstikhin et al., 2018; Prokhorov et al., 2019), and proposing new training strategies (Kim et al., 2018; He et al., 2019). However, most existing work focuses on the VAE framework with Gaussian latent distribution, yet limited work considers the VED framework with normalizing flows. In particular, two questions remain unclear: (1) when the latent distribution is modeled by normalizing flows, does the posterior collapse problem still exist? (2) when posterior collapse exists, what are the appropriate strategies to achieve good summarization quality within the VED framework?

This chapter introduces FlowSUM, a normalizing flows-based VED framework for Transformer-based summarization, along with a controlled alternate aggressive training (CAAT) strategy and a refined gate mechanism to resolve the two challenging issues. Our contributions include:

1. We employ normalizing flows to enrich the latent posterior distribution and integrate the latent code into Transformer-based models in a plug-and-play manner, demonstrating its effectiveness through extensive experiments.

2. We propose a controlled alternate aggressive training strategy and a refined gate mechanism to mitigate the posterior collapse problem and improve training efficacy.

3. Our findings suggest that FlowSUM facilitates knowledge distillation while having a negligible effect on inference time, implying normalizing flows' potential for transferring knowledge from advanced large language models.

4. We investigate the posterior collapse problem for different normalizing flows and examines how the quality of a summary is impacted by the training strategy, gate initialization, and the type and depth of normalizing flows.

This chapter consists of six sections. Section 3.2 provides an overview of normalizing flows, VED, and a summary of related studies. Section 3.3 describes the proposed model architecture and the training strategies employed. Section 3.4 presents the experimental setup and results, and Section 3.5 concludes the chapter with some discussions.

3.2 Backgrounds

3.2.1 Normalizing Flows

Normalizing flows (NF) (Rezende and Mohamed, 2015) are a type of generative model that has gained popularity in recent years. The fundamental idea involves mapping a simple probability density (e.g., Gaussian) to a more complex one through a series of invertible transformations. One of the key advantages of NF is that it allows for exact likelihood evaluations, which is crucial for many applications such as density estimation (Papamakarios et al., 2017), data generation (Tran et al., 2019), and variational inference (Kingma et al., 2016). A flow-based model consists of two components: a base distribution $p_{\mathrm{u}}(\mathbf{u};\psi)$ and a transformation $f(\cdot;\phi):\mathbb{R}^D\rightarrow\mathbb{R}^D$, where f must be invertible and both f and f^{-1} must be differentiable. Let $\mathbf{x}=f(\mathbf{u})$ where $\mathbf{u}\sim p_{\mathrm{u}}(\mathbf{u})$, then the density of $\mathbf{x}$ can be obtained via a change of variables (Bogachev, 2007):

$$\begin{aligned} p_{\mathrm{x}}(\mathbf{x}) &= p_{\mathrm{u}}(\mathbf{u})\left|\det J_f(\mathbf{u})\right|^{-1} \\ &= p_{\mathrm{u}}(f^{-1}(\mathbf{x}))\left|\det J_{f^{-1}}(\mathbf{x})\right|. \end{aligned} \tag{3.1}$$

In this chapter, we examine several NFs, including planar flows (Rezende and Mohamed, 2015), radial flows (Rezende and Mohamed, 2015), Sylvester flows (van den Berg et al., 2018), affine coupling layer (RealNVP by Dinh et al., 2017), inverse autoregressive flow (IAF, Kingma et al., 2016), and neural spline flows (RQNSF by Durkan et al., 2019 and RLNSF by Dolatabadi et al., 2020). We delegate the detailed discussion of transformation and invertibility to Appendix C.5. Throughout the chapter, for each type, we compose K layers of transformation $f_K \circ \cdots \circ f_1(\cdot)$, which remains invertible and differentiable.

3.2.2 Variational Encoder-Decoders

Variational encoder-decoders (VEDs) are generalizations of variational autoencoders (VAEs), and they have been widely used to understand the conditional data generation process. Given (x, y), we assume there exists a latent variable z that follows a certain distribution $p(z; \phi)$ and that y is generated from $p(y|x, z; \theta)$. Since the marginal $p(y|x; \phi, \theta)$ is intractable, we estimate the parameters by maximizing the evidence lower bound (ELBO)[1]:

$$\text{ELBO}_{\text{VED}} = \mathop{\mathbb{E}}_{q(z|x,y)}[\log p(y \mid x, z)] - KL(q(z \mid x, y) \| p(z \mid x)), \tag{3.2}$$

where $q(z \mid x, y)$ is the variational posterior, $p(z \mid x)$ is the latent prior distribution, and $p(y \mid x, z)$ is the encoder-decoder model that generates output conditioned on the input and latent code.

3.2.3 Related Work

Transformer-based Summarization Models

Transformer-based models equipped with pre-training and fine-tuning techniques have enjoyed significant success in many NLP tasks, including text summarization. Liu and Lapata (2019b) proposed BertSUM for extractive and abstractive tasks, utilizing the pre-trained BERT encoder (Devlin et al., 2019). To better align the pre-trained encoder for document understanding with the decoder trained from scratch for text

[1]See derivation in Appendix C.1 Equation (C.1).

generation, Rothe et al. (2020b) demonstrated the effectiveness of leveraging pre-trained BERT (Devlin et al., 2019), GPT-2 (Radford et al., 2019), and RoBERTa (Liu et al., 2019) checkpoints to build sequence-to-sequence (S2S) models for tasks including summarization. Another approach is to address both document understanding and generation in a unified framework by first pre-training some general-purpose S2S models and then fine-tuning on downstream tasks, for instance, BART (Lewis et al., 2020), MASS (Song et al., 2019), UniLM (Dong et al., 2019), ProphetNet (Qi et al., 2020), and T5 (Raffel et al., 2020). In addition, Zhang et al. (2020a) proposed PEGASUS with a pre-training objective tailored for abstractive summarization, achieving significant improvements across multiple datasets.

Variational Summarization

Variational summarization models come in two different flavors. The first models the conditional probability of the target sentences, $p(\mathbf{y} \mid \mathbf{x})$, using a latent variable $\mathbf{z}$ and is represented by $p_\theta(\mathbf{y} \mid \mathbf{x}) = \int p_\theta(\mathbf{z} \mid \mathbf{x}) p_\theta(\mathbf{y} \mid \mathbf{z}, \mathbf{x}) d\mathbf{z}$. The second models the joint probability of both the source and target sentences, $p(\mathbf{x}, \mathbf{y})$, using a latent variable $\mathbf{z}$ and is represented by $p_\theta(\mathbf{x}, \mathbf{y}) = \int p(\mathbf{z}) p_\theta(\mathbf{x} \mid \mathbf{z}) p_\theta(\mathbf{y} \mid \mathbf{z}, \mathbf{x}) d\mathbf{z}$. Our model follows the VED framework and belongs to the first type, similar to previous works such as Setiawan et al. (2020); Fu et al. (2020). Other works, including Zheng et al. (2020); Nguyen et al. (2021); Zou et al. (2021), adopt the second type by jointly modeling topics and sequence-to-sequence generation. Most of them assume a simple Gaussian latent prior, except for Nguyen et al. (2021), which adopts the second type and employs normalizing flows to model neural topic models and enrich global semantics. However, they did not specify the choice of normalizing flows and how they addressed posterior collapse. To the best of our knowledge, there remains limited research on the application of normalizing flows in variational summarization models and their operating characteristics.

3.3 Normalizing Flows Enhanced Summarization

3.3.1 FlowSUM Model Architecture

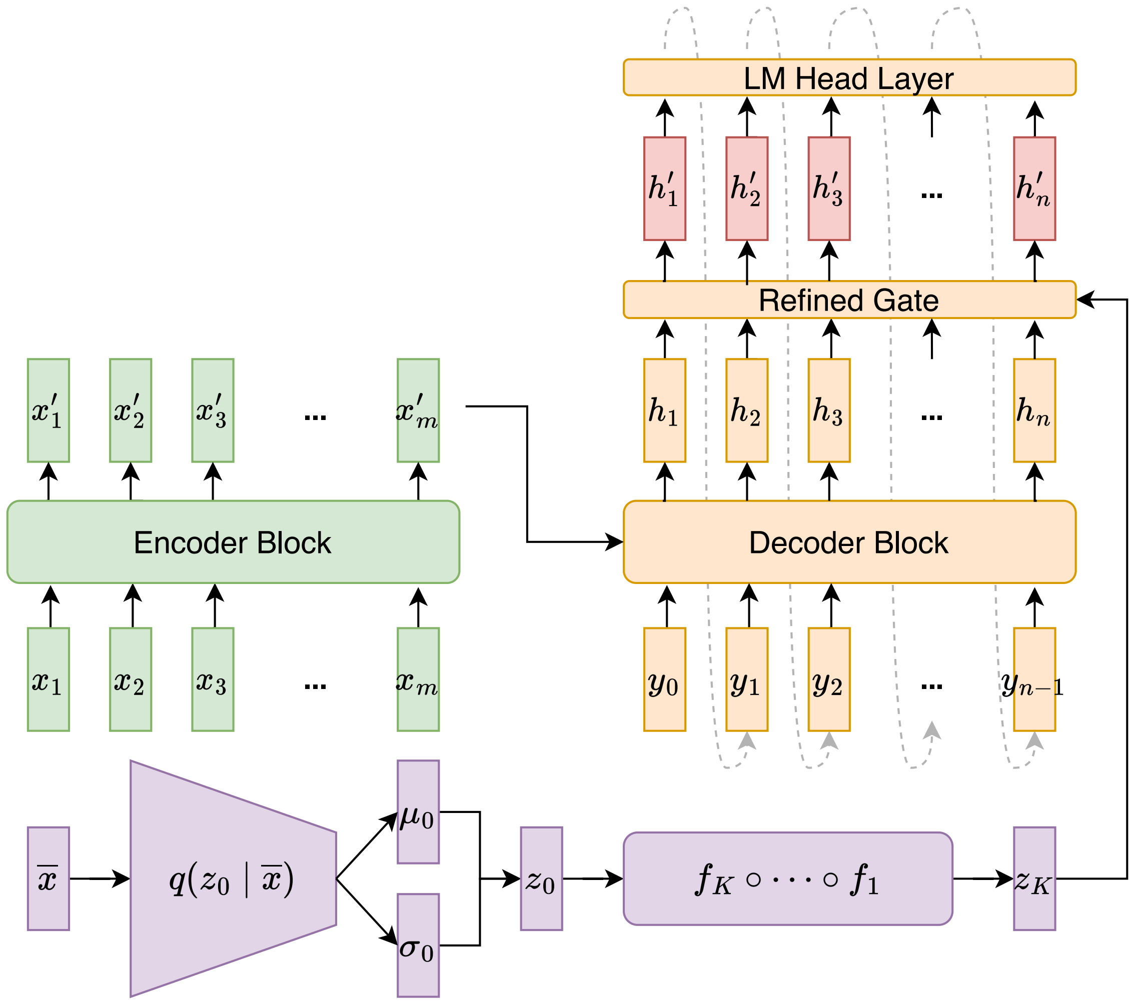
As illustrated in Figure 3.1, FlowSUM consists of three key components: an NF latent module, a Transformer-based encoder-decoder, and a refined gate mechanism. Throughout this section, let e be the embedding size, V be the vocabulary size, m, n be the length of the input source and target summary respectively, ℓ be the latent dimension of the NF latent module, and d be the dimension of the decoder's hidden states. Let $\mathbf{E} = \{E_v\}_{v=1}^V$ be the input embedding vectors, $\boldsymbol{x} = \{x_i\}_{i=1}^m$ be the input source text, $\boldsymbol{y} = \{y_j\}_{j=1}^n$ be the target summary text, and $\boldsymbol{b} = \{b_v\}_{v=1}^V$ be the Bag-of-Words (BoW) representation of the input source text[2].

NF Latent Module. The NF latent module comprises of an inference network $q(\cdot)$ and a normalizing flows model. The inference network learns the means and standard deviations of the distribution of z_0, which is assumed to be Gaussian. It takes the average embedding of the untruncated input source text $\overline{x} = \frac{\sum_{v=1}^V b_v E_v}{\sum_{v=1}^V b_v} \in \mathbb{R}^e$ as input and outputs $\mu_0 \in \mathbb{R}^\ell$ and $\sigma_0 \in \mathbb{R}^\ell$[3]. Subsequently, a random sample z_0 is drawn from $z_0 \sim N(\mu_0, \Sigma_0) \in \mathbb{R}^\ell$, where Σ_0 is a diagonal matrix with σ_0^2 on the diagonal. Finally, the normalizing flows model applies K layers of invertible transformations to z_0 to obtain the latent code $z_K = f_K \circ \cdots \circ f_1(z_0) \in \mathbb{R}^\ell$.

Gated Transformer-based Encoder-Decoder. Our model follows the Transformer-based encoder-decoder framework. The encoder processes the input text and learns a sequence of hidden representations, and the decoder generates a summary based on the encoder's hidden states and the previously generated tokens. We incorporate the latent information into the decoder with a gate mechanism, which mixes the latent

[2]When we don't truncate the input text, $b^T \mathbb{1} = m$ holds. However, if we truncate the input due to encoder constraints, then $b^T \mathbb{1} > m$, and the BoW vector will contain information that would otherwise have been lost.

[3]In the code, $\log(\sigma_0)$ is used as output since its range is $(-\infty, \infty)$.



LM Head Layer
h'_1 h'_2 h'_3 ... h'_n
Refined Gate
x'_1 x'_2 x'_3 ... x'_m
h_1 h_2 h_3 ... h_n
Encoder Block
Decoder Block
x_1 x_2 x_3 ... x_m
y_0 y_1 y_2 ... y_{n-1}
$\overline{x}$
$q(z_0 \mid \overline{x})$
μ_0
σ_0
z_0
$f_K \circ \cdots \circ f_1$
z_K

Figure 3.1: FlowSUM Model Architecture.

vector z_K with the decoder's last hidden layer's hidden states $\{h_j\}_{j=1}^n$. As pointed out in Gu et al. (2020), the saturation property of traditional gating mechanisms hinders gradient-based optimization. Therefore, following their proposal, we use a refined gate mechanism designed to allow for better gradient flow. Let $\sigma(\cdot)$ be the sigmoid function. Then we generate the gated fused hidden states $\{h'_j\}_{j=1}^n$ as in Equation (3.3).

$$\begin{aligned}
&z'_K = W^z z_K \in \mathbb{R}^d, \text{ where } W^z \in \mathbb{R}^{d\times \ell} \\
&f_j = \delta\left(W^f\left[h_j; z'_K\right]\right) \in \mathbb{R}^d, \text{ where } W^f \in \mathbb{R}^{d\times 2d} \\
&r_j = \delta\left(W^r\left[h_j; z'_K\right]\right) \in \mathbb{R}^d, \text{ where } W^r \in \mathbb{R}^{d\times 2d} \\
&g_j = (1-r_j)\cdot f_j^2 + r_j\left(1-(1-f_j)^2\right) \in \mathbb{R}^d \\
&h'_j = (1-g_j)\cdot h_j + g_j \cdot z'_K \in \mathbb{R}^d
\end{aligned} \tag{3.3}$$

Afterward, the fused hidden states are passed to a language model (LM) Head layer, where they are transformed into vectors modeling the probabilities of each word in the vocabulary.

3.3.2 Training Objective

As discussed in Section 3.2.2, we estimate parameters by maximizing the ELBO. We follow Zhou and Neubig (2017) and assume all the information in y is contained in x and hence $q(z \mid x, y) = q(z \mid x)$. Traditional VEDs assume $q(z \mid x)$ to be a Gaussian, and the conditional prior $p(z \mid x)$ is $N(0, I)$, allowing analytical computation of the KL term and gradient-based optimization using the reparameterization trick (Kingma and Welling, 2014). However, in a normalizing flows-based VED with K layers of transformations $f_K \circ \cdots \circ f_1$, $q(z \mid x) = q(z_K \mid x)$ can be complex, and the KL term lacks an analytical form. Therefore, we rewrite the ELBO in Equation (3.2) via a change of variables[4]:

$$
\begin{aligned}
&\mathrm{ELBO}_{\text{NF-VED}} \\
=&\mathbb{E}_{q_0(z_0)}\Big[\log p\left(y \mid x, z_K\right)+\log p\left(z_K \mid x\right)\Big] \\
&-\mathbb{E}_{q_0(z_0)}\left[\log q_0\left(z_0\right)-\sum_{k=1}^{K} \log \left|\det J_{f_k}\left(z_{k-1}\right)\right|\right],
\end{aligned} \tag{3.4}
$$

where q_0 and q_K are the probability density function for z_0 and z_K respectively.

Let $\mathcal{L}_{\mathrm{CE}}$ denote the cross-entropy loss and $\mathcal{L}_{\mathrm{VI}}$ denote the loss introduced by the variational latent module. We then use the idea of Monte Carlo to derive the training objective as in Equation (3.5). Note that $\mathcal{L}_{\mathrm{VI}}$ is a Monte Carlo estimate of the KL divergence between q_K and the prior distribution $p(z_K \mid x)$. Here, we assume that the prior is a standard Gaussian distribution, as in Setiawan et al. (2020).

$$
\begin{aligned}
\mathcal{L} =&\mathcal{L}_{\mathrm{CE}}+\mathcal{L}_{\mathrm{VI}} \\
=&-\sum_{j=1}^{n} \log p\left(y_j \mid \left\{x_i\right\}_{i=1}^{m}, z_K, y_{<j}\right)+\log q_0\left(z_0\right) \\
&-\sum_{k=1}^{K} \log \left|\det J_{f_k}\left(z_{k-1}\right)\right|-\log p\left(z_K \mid x\right)
\end{aligned} \tag{3.5}
$$

3.3.3 Mitigating Posterior Collapse

To remedy posterior collapse, we consider two strategies, aiming to preserve the expressiveness of the latent variable and improve the overall summary quality. The first approach, called β_C-VAE (Prokhorov et al., 2019), replaces the KL term with $\beta|KL - C|$, where β is a scaling factor, and $C \geq 0$ is a threshold that regulates the magnitude of the KL term. When $C > 0$, the KL term is expected to be discouraged from getting close to 0.

[4]See derivation in Appendix C.1 Equation (C.2).

We propose the second approach, Controlled Alternate Aggressive Training (CAAT), inspired by the lagging inference strategy (He et al., 2019). This strategy uses the observation that the inference network cannot accurately approximate the true posterior in the initial stages of training. As outlined in Algorithm 1, CAAT comprises two stages. In the first stage, we alternately update the variational parameters and the entire parameters[5] for a specified number of steps. In the second stage, we train all parameters jointly, as in basic VAE training, for the remainder of the training.

Algorithm 1 Controlled Alternate Aggressive Training (CAAT)

```
Input: number of aggressive training steps $n_{agg}$; maximum number of training steps $n_{max}$; number of alternating steps $n_{alt}$.
1: $\boldsymbol{\theta}, \boldsymbol{\psi} \leftarrow$ Initialize model parameters and variational parameters respectively
2: for $i = 1, 2, \cdots, n_{agg}$ do
3:     $\mathbf{X} \leftarrow$ Random data minibatch
4:     if $i \mod n_{alt} = 0$ then
5:         Compute $\boldsymbol{g}_{\boldsymbol{\theta},\boldsymbol{\psi}} \leftarrow \nabla_{\boldsymbol{\psi},\boldsymbol{\theta}} \mathcal{L}(\mathbf{X}; \boldsymbol{\theta}, \boldsymbol{\psi})$
6:         Update $\boldsymbol{\theta}, \boldsymbol{\psi}$ using gradients $\boldsymbol{g}_{\boldsymbol{\theta},\boldsymbol{\psi}}$
7:     else
8:         Compute $\boldsymbol{g}_{\psi} \leftarrow \nabla_{\boldsymbol{\psi}} \mathcal{L}(\mathbf{X}; \boldsymbol{\theta}, \boldsymbol{\psi})$
9:         Update $\boldsymbol{\psi}$ using graidents $\boldsymbol{g}_{\boldsymbol{\psi}}$
10: for $i = n_{agg}, n_{agg} + 1, \cdots, n_{max}$ do
11:     $\mathbf{X} \leftarrow$ Random data minibatch
12:     Compute $\boldsymbol{g}_{\boldsymbol{\theta},\boldsymbol{\psi}} \leftarrow \nabla_{\boldsymbol{\psi},\boldsymbol{\theta}} \mathcal{L}(\mathbf{X}; \boldsymbol{\theta}, \boldsymbol{\psi})$
13:     Update $\boldsymbol{\theta}, \boldsymbol{\psi}$ using gradients $\boldsymbol{g}_{\boldsymbol{\theta},\boldsymbol{\psi}}$
14:     if early stopping criterion is met then
15:         break
```

The CAAT strategy offers another benefit by enabling greater training control. While it is often thought that giving a model enough freedom to learn, even without the aid of the NF module, will not have a detrimental effect on performance, our experiments demonstrate that this assumption is not valid, especially for datasets

[5]In our preliminary experiments, we find that if we alternate between variational and encoder-decoder parameters, the training becomes unstable and generates NaN values. Therefore, we alternate between variational and all parameters.

with short summaries. In such cases, the model will not learn on its own to avoid harming the original performance. By setting n_{agg} and n_{alt} to high values, the CAAT strategy allows us to effectively freeze the encoder-decoder parameters, ensuring that when the nf module is unhelpful, it will not significantly impact performance.

3.3.4 NF-enhanced Knowledge Distillation

Normalizing flows can learn complex and multi-modal distributions (Papamakarios et al., 2017), which makes them a promising approach for knowledge distillation tasks that involve integrating information from multiple sources (Hinton et al., 2015). To investigate the impact of normalizing flows on knowledge distillation, we adopt two knowledge distillation methods by Shleifer and Rush (2020): Shrink and Fine-Tune (SFT) and Pseudo-labels (PL). SFT shrinks the teacher model and re-finetunes the shrunk model. In contrast, the PL method initializes the student model with the compressed version produced by SFT and then fine-tunes on the pseudo-labeled data generated by the teacher model. In this study, we fine-tune the model on the augmented data with both original and pseudo-labeled data, enabling it to more effectively switch between generated summaries and ground truth, thereby mitigating exposure bias.

3.4 Experiments

3.4.1 Datasets

We evaluate the effectiveness of FlowSUM on six public benchmark datasets[6] as follows. These datasets exhibit various summary styles and lengths, and their corresponding statistics are shown in Table 3.1.

[6] We access them through Hugging Face Datasets, which provides reproducible code for processing texts and generating train/validation/test splits.

Datasets	Split (train/val/test)	Avg. doc length	Avg. summary length
CNN/DM	287113/13368/11490	781	56
Multi-News	44972/5622/5622	2103	264
arXiv	203037/6436/6440	4938	220
PubMed	119924/6633/6658	3016	203
XSum	204045/11332/11334	431	23
SAMSum	14732/818/819	94	20

Table 3.1: Statistics of Summarization Datasets.

CNN/Daily Mail (Hermann et al., 2015) consists of 312,085 online news articles, with one article paired with a multi-sentence summary. We use the non-anonymized version as in See et al. (2017a) and follow the text processing[7] in Lewis et al. (2020).

Multi-News (Fabbri et al., 2019a) is a multi-document dataset comprising 56k pairs of news articles and multi-sentence summaries.

arXiv, PubMed (Cohan et al., 2018) are two scientific paper document datasets from arXiv.org (113k) and PubMed (215k). Each pair consists of a scientific article's body document and its abstract.

XSum (Narayan et al., 2018a) contains 227k BBC articles, each summarized in a single sentence.

SAMSum (Gliwa et al., 2019) includes 16k conversations annotated with summaries by linguists. Unlike structured texts, the information in dialogues is scattered across different speakers' utterances, increasing the summarization difficulty.

3.4.2 Implementation Details

We configure the inference net $q(z_0|\overline{x})$ to be a feedforward neural network and set the latent dimension ℓ to 300 and the number of NF layers $\in \{2, 4, 6, 8\}$. For models

[7] We update the data loading script following `https://github.com/facebookresearch/fairseq/issues/1401` and publish the updated data loading script at Hugging Face Hub.

that use β_C-VAE, we set $\beta = 1$ and $C = 0.1$, and for models that use CAAT, we conduct one epoch of aggressive training with $n_{alt} = 15$, followed by two epochs of non-aggressive training. See more details in Appendix C.2.

3.4.3 Baselines

We use BART (Lewis et al., 2020) and BERT2BERT (Rothe et al., 2020b) as two backbone models. We refer to the VAE-based versions as VAESUM[8] and the PL knowledge distilled FlowSUM as FlowSUM-PLKD. We compare them with the following baselines.

PG+Cov (See et al., 2017a) is a pointer-generator (PG) network supplemented with a coverage mechanism that addresses the Out-Of-Vocabulary problem and minimizes word repetition.

BERT2BERT (Rothe et al., 2020b) initializes both the encoder and the decoder with the pre-trained BERT checkpoints and adds cross-attention layers.

BERTSUM (Liu and Lapata, 2019b) builds on top of BERT and applies a fine-tuning scheduler to better align the encoder and the decoder.

BART (Lewis et al., 2020) is a pretrained denoising autoencoder with the standard sequence-to-sequence Transformer architecture. In this chapter, we use BART as the encoder-decoder backbone.

PEGASUS (Zhang et al., 2020a) is a large Transformer-based S2S model, pre-trained on massive text data using a self-supervised objective called gap sentence generation, designed for abstractive summarization.

VHTM (Fu et al., 2020) is a variational hierarchical model built on the PG network. It models the topic proportion vector with isotropic Gaussian and fuses in topic information at diverse granularity levels.

[8]VAESUM is a special case of FlowSUM with zero layers of normalizing flows transformation.

TAS (Zheng et al., 2020) is a topic-guided Transformer-based S2S model that injects the topic-word matrix into the LMHead layer and jointly trains the NTM and encoder-decoder model.

PEGASUS+Flow-NTM (Nguyen et al., 2021) is a topic-aware model built on PEGASUS. It utilizes a Flow-based NTM and a contextualized gating mechanism to integrate topic information into the encoder and the decoder.

3.4.4 Results

Automatic Evaluation

We evaluate the generated summary quality using ROUGE scores (Lin, 2004b) and BERTScore (Zhang et al., 2020b)[9]. Specifically, we utilize the overlap of unigrams and bigrams (ROUGE-1 and ROUGE-2) to evaluate the informativeness, and the longest common subsequence (ROUGE-L) for fluency. Moreover, we report BERTScore, which gauges semantic similarity based on contextual embeddings. Furthermore, we present rep-w (Fu et al., 2021)[10] and the average length of summaries to gain a better understanding of the quality.

We compare the proposed model against baseline models in ROUGE scores in Table 3.2 and Table 3.3. On CNN/DM, FlowSUM (BERT2BERT) greatly outperforms BERT2BERT, whereas VAESUM adds noise to the model and leads to a decrease in performance. With the BART backbone, FlowSUM achieves an absolute improvement over the BART model with +0.48, +0.08, and +0.75 in R-1, 2, and L scores, respectively. However, on XSum, the variational models do not perform well when the gold summaries involve only one sentence. VAESUM leads to a significant decrease in performance, whereas with FlowSUM, the decrease in ROUGE scores is less severe,

[9]We obtain both metrics using Hugging Face Evaluate and report the F_1 scores.

[10]rep-w is calculated as the proportion of the current token that appears in the previous w tokens. Refer to Equation (2.3) for the detailed definition.

Model	ROUGE ↑		
	1	2	L
PG+Cov (See et al., 2017a)	39.53	17.28	36.38
BERT2BERT (Rothe et al., 2020b)	41.28	18.69	38.09
BERTSUM (Liu and Lapata, 2019b)	42.13	19.60	39.18
BART (Lewis et al., 2020)	44.16	21.28	40.90
PEGASUS (Zhang et al., 2020a)	44.17	21.47	41.11
VHTM (Fu et al., 2020)	40.57	18.05	37.18
TAS (Zheng et al., 2020)	44.38	21.19	41.33
PEGASUS+NTM (Nguyen et al., 2021)	44.52	**21.95**	41.39
VAESUM (BERT2BERT)	40.89	18.28	37.95
FlowSUM (BERT2BERT)	41.51	18.81	38.56
VAESUM (BART)	44.36	21.09	41.37
FlowSUM (BART)	**44.64**	21.36	**41.65**
FlowSUM-PLKD (BART)	44.59	21.49	41.59

Table 3.2: Comparison with baselines on CNN/DM.

leading to +0.12, -0.15, and -0.25 in R-1, 2, and L scores, respectively.

Table 3.4 uses BART as the backbone and compares BART, VAESUM, and FlowSUM across all datasets. Overall, variational models produce summaries of superior quality for datasets with long summaries, such as CNN/DM, Multi-News, arXiv, and PubMed, and FlowSUM further enhances the performance beyond VAESUM. However, when it comes to datasets featuring short summaries such as XSum and SAMSum, the variational component markedly diminishes the model performance. We hypothesize that brief summaries may be more susceptible to disturbances and are more prone to being affected by noise. Nevertheless, incorporating NF modules alleviates these reductions and accomplishes comparable outcomes. Furthermore, we observe that both variational models tend to generate lengthier summaries, while FlowSUM exhibits fewer issues with repetition compared to VAESUM.

Model	ROUGE ↑		
	1	2	L
PG+Cov (See et al., 2017a)	28.10	8.02	21.72
BERTSUM (Liu and Lapata, 2019b)	38.81	16.50	31.27
BART (Lewis et al., 2020)	45.14	22.27	37.25
PEGASUS (Zhang et al., 2020a)	47.21	24.56	39.25
TAS (Zheng et al., 2020)	44.63	21.62	36.77
PEGASUS+NTM (Nguyen et al., 2021)	**49.57**	**25.08**	**41.81**
VAESUM (BART)	43.62	20.27	35.06
FlowSUM (BART)	45.26	22.12	37.00
FlowSUM-PLKD (BART)	45.54	22.67	37.38

Table 3.3: Comparison with baselines on XSum.

On NF-enhanced Knowledge Distillation

We use PEGASUS as the teacher model to generate pseudo-labels on the CNN/DM training set. In this study, we explore the effects of knowledge distillation on BART and DistilBART, a shrunken version of BART. We examine two variations of DistilBART: dBART-6-6, which replicates 6 layers[11] of the BART encoder and decoder, and dBART-12-3, which duplicates all layers of the BART encoder and 3 layers[12] of the decoder.

Table 3.5 presents the impact of the PL approach on the original BART model. Training the BART model on augmented data worsens the performance compared to training on the original data. In contrast, VAESUM-PLKD achieves improvements in all three ROUGE scores, and FlowSUM-PLKD with RQNSF achieves the highest R-2 score, albeit with some sacrifice in R-1 and R-L[13]. However, planar flows appear to be unsuitable for knowledge distillation via PL. To better understand FlowSUM-PLKD, we visualize the latent distribution (see Appendix C.4) and demonstrate how the NF's

[11] The 0, 2, 4, 7, 9, and 11th layer.

[12] The 0, 6, and 11th layer.

[13] This can be explained by the teacher model's worse performance in these two metrics.

Model	ROUGE ↑ 1/2/L	BERT-Score ↑	rep-w ↓	Length
CNN/DM				
BART	44.16/21.28/40.90	89.40	**8.31**	84.11
VAESUM	44.34/21.09/41.37	89.20	8.43	88.63
FlowSUM	**44.64**/**21.36**/**41.65**	**89.46**	8.43	92.24
Multi-News				
BART	42.56/15.34/36.67	86.69	**9.76**	133.42
VAESUM	43.91/16.68/38.10	87.04	9.95	128.79
FlowSUM	**44.42**/**17.01**/**38.36**	**87.09**	9.91	128.87
arXiv				
BART	42.55/15.92/37.89	85.35	17.23	130.68
VAESUM	43.05/**16.34**/38.26	85.44	16.63	130.92
FlowSUM	**43.11**/16.26/**38.31**	**85.45**	**16.55**	132.88
PubMed				
BART	41.57/16.72/36.94	84.65	13.26	136.10
VAESUM	44.21/19.20/39.32	85.07	12.76	138.70
FlowSUM	**44.55**/**19.50**/**39.59**	**85.16**	**12.59**	138.09
XSum				
BART	45.14/**22.27**/**37.25**	**92.16**	**4.63**	25.54
VAESUM	43.62/20.27/35.06	91.75	5.96	31.22
FlowSUM	**45.26**/22.12/37.00	92.13	4.95	28.71
SAMSum				
BART	**53.16**/28.19/**49.03**	**92.68**	6.71	30.00
VAESUM	51.91/26.74/47.41	92.40	7.53	30.92
FlowSUM	53.13/**28.49**/49.00	92.67	**6.59**	29.77

Table 3.4: Comparison of BART, VAESUM (BART), and FlowSUM (BART) on all six benchmarks.

Model	ROUGE ↑			BERT-Score ↑	Length
	1	2	L		
BART	44.16	21.28	40.90	89.40	84.11
VAESUM	44.34	21.09	41.37	89.20	88.63
FlowSUM (Planar)	44.62	21.32	41.64	89.20	90.78
FlowSUM (RQNSF)	**44.64**	21.36	**41.65**	89.46	92.24
PEGASUS	44.17	21.47	41.11	**89.52**	77.84
BART-PLKD	42.83	20.16	39.98	89.04	100.52
VAESUM-PLKD	44.45	21.25	41.45	89.41	93.42
FlowSUM-PLKD (Planar)	44.19	21.03	41.15	89.34	92.38
FlowSUM-PLKD (RQNSF)	44.59	**21.48**	41.59	89.47	84.75

Table 3.5: PL Knowledge Distillation on BART on CNN/DM.

ability to capture multi-modality could account for its impressive performance.

Table 3.6 investigates the two DistilBART variants with RQNSF. With FlowSUM, both variants achieve improvements, suggesting that NF is beneficial for the SFT approach. Previous experiments from Shleifer and Rush (2020) showed that PL performed worse than SFT on CNN/DM. However, our experiments reveal that the NF latent module unleashes the potential of PL. When trained on augmented data, FlowSUM-PLKD (dBART-6-6) achieves R-1/2/L improvements of 0.92/0.47/1.01 over dBART-6-6, and FlowSUM-PLKD (dBART-12-3) achieves improvements of 0.66/ 0.49/0.63 over dBART-12-3, much more than the SFT approach. Furthermore, FlowSUM does not introduce additional computational burden at inference, and the time cost is primarily related to the length of the generated summaries.

Analysis on NF Types and Depth

We investigate the effect of NF types and the number of NF layers on the Multi-News dataset[14]. Table 3.7 explores the effect of NF types. Simple flows like Planar

[14]We choose Multi-News because of its smaller size, which allows for experiments to be completed at a lower computational cost.

Model	ROUGE ↑ 1/2/L	BERT-Score ↑	Length	# Params (MM)	Inference Time (MS) ↓
dBART-6-6					
dBART-6-6	42.78/20.24/39.72	88.98	67.42	230	170.5
FlowSUM	43.41/20.33/40.41	89.18	91.25	238	234.9
FlowSUM-PLKD	**43.70**/**20.71**/**40.73**	**89.24**	91.10	238	239.7
dBART-12-3					
dBART-12-3	43.39/20.57/40.44	89.20	85.48	255	199.6
FlowSUM	43.53/20.61/40.59	89.28	83.74	263	190.7
FlowSUM-PLKD	**44.05**/**21.06**/**41.07**	**89.37**	84.48	263	200.4

Table 3.6: Knowledge Distillation on DistilBART on CNN/DM.

Model	ROUGE ↑ 1/2/L	BERT-Score ↑	rep-w ↓	Length
BART	42.56/15.35/36.67	86.69	**9.76**	133.42
VAESUM	43.91/16.68/38.10	87.04	9.95	128.79
FlowSUM (Planar)	43.85/16.61/37.97	87.03	10.04	128.84
FlowSUM (Radial)	43.84/16.68/37.98	87.04	9.92	128.72
FlowSUM (Sylvester)	44.18/16.71/38.15	87.08	9.80	128.76
FlowSUM (RealNVP)	44.19/16.64/38.15	87.05	9.81	128.76
FlowSUM (IAF)	**44.42**/**17.01**/**38.36**	**87.09**	9.91	128.87
FlowSUM (RLNSF)	44.25/16.86/38.14	87.06	9.80	128.80
FlowSUM (RQNSF)	44.31/16.98/38.27	87.07	9.91	128.81

Table 3.7: Effect of NF Types on Multi-News.

and Radial yield inferior performance compared to the VAE counterpart, whereas more complex flows tend to achieve greater improvements. Overall, IAF and RQNSF emerge as the best-performing NF types.

Table 3.8 delves further into IAF and RQNSF, investigating the effect of NF depth. The findings indicate that adding more layers does not always lead to improved performance. We hypothesize that when the encoder-decoder model is well-trained, the increased complexity of the NF module may introduce more noise, outweighing the benefits of better latent modeling and subsequently worsening the summary quality.

Model	ROUGE ↑ 1/2/L	BERT-Score ↑	rep-w ↓	Length
FlowSUM (IAF-4)	44.30/**17.03**/38.22	87.05	**9.82**	128.81
FlowSUM (IAF-6)	**44.42**/17.01/**38.36**	**87.09**	9.91	128.87
FlowSUM (IAF-8)	44.18/16.90/38.16	87.04	9.88	128.84
FlowSUM (RQNSF-2)	44.15/16.88/38.20	87.04	9.94	128.83
FlowSUM (RQNSF-4)	**44.31**/**16.98**/**38.27**	**87.07**	9.91	128.81
FlowSUM (RQNSF-6)	44.15/16.88/38.18	87.06	**9.87**	128.92

Table 3.8: Effect of Number of NF Layers on Multi-News.

Analysis on Training Strategies

We implement standard VAE training, β_C-VAE, and CAAT on VAESUM and FlowSUM models, and we evaluate their effectiveness with different types of normalizing flows. Table 3.9 shows that VAESUM and FlowSUM models with residual flows, including planar, radial, and Sylvester flows, suffer from posterior collapse, whereas those with more complex flows do not. Moreover, applying β_C-VAE to VAESUM and FlowSUM models with residual flows does not effectively mitigate posterior collapse but even exacerbates the issue. Furthermore, for models with planar, RealNVP, and IAF flows, training with β_C-VAE worsens ROUGE scores, while for radial and Sylvester flows, it improves performance. Notably, the two neural spline flows are not impacted by β_C-VAE training.

Concerning CAAT, we note that applying it to treat severe posterior collapses such as VAESUM and FlowSUM with residual flows can cause instability in training while producing NaN values. Hence, it is only effective for models with KL divergence that is not close to zero. Nonetheless, when applicable, CAAT enhances the quality of summaries, particularly when utilized with the top-performing NFs, namely IAF and RQNSF.

In addition, we explore the impact of gate score initialization. The standard

Model	Training	ROUGE ↑			KL
		1	2	L	Divergence
VAESUM	standard	43.91	16.68	38.10	0.0117
VAESUM	β_C-VAE	43.78	16.54	37.96	0.0082
FlowSUM (Planar)	standard	43.85	16.61	37.97	0.2719
FlowSUM (Planar)	β_C-VAE	43.68	16.47	37.85	0.1815
FlowSUM (Radial)	standard	43.63	16.37	37.82	0.0121
FlowSUM (Radial)	β_C-VAE	43.84	16.68	37.98	0.0096
FlowSUM (Sylvester)	standard	43.68	16.51	37.87	0.0841
FlowSUM (Sylvester)	β_C-VAE	44.18	16.71	38.15	0.0348
FlowSUM (RealNVP)	standard	44.19	16.64	38.15	4.7986
FlowSUM (RealNVP)	β_C-VAE	43.71	16.54	37.85	7.8938
FlowSUM (RealNVP)	CAAT	44.12	16.82	38.11	5.2107
FlowSUM (IAF)	standard	43.87	16.62	37.97	3.9146
FlowSUM (IAF)	β_C-VAE	43.81	16.58	37.91	3.9128
FlowSUM (IAF)	CAAT	44.30	17.03	38.22	2.1108
FlowSUM (RLNSF)	standard	44.25	16.86	38.14	104.9667
FlowSUM (RLNSF)	β_C-VAE	44.25	16.86	38.14	104.9667
FlowSUM (RLNSF)	CAAT	44.14	16.82	38.05	95.3774
FlowSUM (RQNSF)	standard	44.18	16.76	38.18	127.8106
FlowSUM (RQNSF)	β_C-VAE	44.18	16.76	38.18	127.8106
FlowSUM (RQNSF)	CAAT	44.31	16.98	38.27	107.0794

[a] VAESUM and FlowSUM with radial flows have no CAAT results as the training is unstable and generates NaN values.

Table 3.9: Effect of Training Strategies.

Training	Gate Initialization	ROUGE ↑		
		1	2	L
standard	standard	40.82	18.29	37.92
standard	near-zero	40.98	18.36	38.09
CAAT	standard	**41.51**	**18.81**	**38.56**
CAAT	near-zero	41.13	18.57	38.21

Table 3.10: Effect of CAAT and Gate Initialization.

method initializes gating weights with small deviations from zero, resulting in an initial gate score close to 0.5. In contrast, the near-zero initialization method initializes gating weights such that the resulting gate score is approximately 0.05. Our experiments using FlowSUM (BERT2BERT) with RQNSF as the base model reveal that CAAT + Standard Gate Score Initialization yields the best results and the most stable training process, as illustrated in Table 3.10 and Figure C.1 to Figure C.2 in Appendix C.3. This suggests that by setting a large initial gate score and forcing the model to learn from the NF latent module, we can better capture latent code information.

3.5 Conclusions and Discussions

This chapter introduces FlowSUM, a normalizing flows-based Variational Encoder-Decoder (VED) framework for text summarization. It outperforms a leading non-latent model across multiple datasets. This enhanced performance is attributed to the flexible posterior distributions provided by normalizing flows. We also analyze the operating characteristics and the posterior collapse problem of normalizing flows and propose an effective training strategy for complex flows. Moreover, we demonstrate that incorporating normalizing flows is highly effective for knowledge distillation with minimal impact on inference time.

FlowSUM illustrates the advantages of incorporating flexible latent modeling. Considering the remarkable achievements of Latent Diffusion Models (LDMs) in generating images (Rombach et al., 2022), adopting LDMs for capturing latent representation may produce comparable or even superior outcomes in text summarization. In this scenario, the gating mechanism may not be an appropriate choice. A direct correlation between the latent vector and the target text may be more suitable for executing the diffusion process. Enhancing the architecture to leverage diffusion models could be a potential avenue for future research.

3.6 Limitations

FlowSUM has demonstrated excellent results on datasets with long summaries. However, its performance on short-summary datasets like XSum and SAMSum has been unsatisfactory. The underlying cause could be attributed to suboptimal hyperparameter tuning or the incompatibility of FlowSUM with short summaries. Additional investigations are needed to identify the root cause.

Furthermore, we did not fine-tune the hyper-parameters of the normalizing flows model, such as the latent dimension, the number of bins in spline coupling layers, and the neural network in IAF, RealNVP, RLNSF, and RQNSF. Adjusting these hyper-parameters could potentially enhance the model's performance.

Due to limited computational resources, we utilized BART and BERT2BERT as the backbone models instead of newer architectures. Further research may focus on verifying the effectiveness of FlowSUM on more advanced structures.

Part II

Causal Discovery

Chapter 4

A Hierarchical Ensemble Causal Structure Learning Approach for Wafer Manufacturing

4.1 Introduction

In modern manufacturing, the production systems become increasingly automated yet complex to achieve higher product quality and diversity (Liang et al., 2004). However, an increased complexity imposes challenges for understanding the underlying causal mechanism of a production line and identifying the root causes of system failures and product defects. In such a situation, gaining knowledge of causal relations between components in the assembly line is crucial. It enhances engineers' understanding while permitting the root-cause tracing of a failure event to allow real-time error corrections in the absence of on-site engineers. Moreover, the causal relations provide precautionary warnings to potential future errors (Huegle et al., 2020), which reduces the chance of assembly line shutdown.

A standard practice in manufacturing is to learn causal relationships through the design of experiments, which is very costly and time-consuming given the thousands of factors examined in a production environment. However, recent advances in sensors and automatic measurement tools open the door to causal discovery, which is about learning causal relations from observational data (Spirtes et al., 2000; Pearl, 2009). As a result, much progress has been made in applying causal discovery in the manufacturing domain, such as fault propagation analysis on industrial board machines (Landman and Jämsä-Jounela, 2016), failure precaution in automotive body production (Huegle et al., 2020), and root cause diagnosis in fluid catalytic cracking unit (Gharahbagheri et al., 2015) and semiconductor manufacturing (Shah et al., 2018).

This chapter focuses on learning the causal relations in the wafer manufacturing domain and, more specifically, the causal relations among the sensors and abnormal events in a wafer assembly line. We identify several notable challenges regarding causal discovery through a case study on a wafer assembly line from Seagate Technology. First, to help with error tracing, we need to learn the causal structure among the sensors and the abnormal events from all steps, which includes tens of thousands of variables and leads to a high-dimensional scenario under which the scalability and theoretical consistency may not be guaranteed (Nandy et al., 2018; Colombo et al., 2014). Second, causal discovery methods designed for a specific type may not apply as data come from multiple sources, and methods designed for mixed data (Andrews et al., 2018; Cui et al., 2016) are usually computationally inefficient in high-dimensional scenarios. Third, abnormal events are rare in practical production, and as suggested in our simulations (in Table 4.2), data imbalance would significantly impact the accuracy of causal discovery. This problem has not received full attention in the research area. Barnes et al. (2019) and Runge et al. (2019) identified the class imbalance as a challenge in causal discovery but did not propose any effective methods in response. Fourth, the products often do not strictly follow the prespecified procedure in real-life production. Such procedure deviations lead to irregularly missing data and few observations in the merged data across all steps, upon which it is impossible to learn a causal structure. Existing approaches for missing data (Tu et al., 2019; Gao et al.,

2022) are either computationally expensive or incapable of handling mixed data. Finally, the manufacturing process imposes temporal and domain constraints on the causal structure. Properly incorporating these constraints helps reduce the candidate causal graph complexity and helps avoid factual errors and hence requires deliberate attention.

To address these challenges, we propose a hierarchical ensemble approach that leverages temporal and domain constraints on the assembly line and quantifies the uncertainty of causal discovery. The approach consists of three phases: (1) block-level learning, in which we cluster the steps on the assembly line into blocks and learn a block-level structure with distilled constraints; (2) step-level learning, in which we learn constrained causal structures at a finer granularity based on the block-level structure; and (3) aggregation, where we aggregate the step-level structures and quantify the uncertainty of the learned relations. Our approach offers several advantages. First, it is scalable and can handle high-dimensional data with mixed types. Second, it addresses imbalanced data due to rare events and can handle irregular missing patterns. Finally, it incorporates domain and temporal knowledge to refine the candidate graphs, increasing the accuracy of causal discovery and avoiding factual errors. We demonstrate the effectiveness of our approach through simulations and an application to the wafer manufacturing data from Seagate Technology. The causal structure learned from the data is cross-validated by domain experts, and our proposed modeling pipeline and visualizing tool have received positive feedback from engineers and technicians.

The chapter consists of six sections. Section 4.2 describes the data characteristics in the wafer manufacturing domain and discusses related work and challenges. Section 4.3 introduces causal structure learning backgrounds, and Section 4.4 illustrates the proposed methods. Section 4.5 performs simulations to investigate the operating characteristics of the proposed methods and applies the methods to analyze a real data set from Seagate Technology. Finally, we conclude the chapter with some discussions.

4.2 Manufacturing Data, Related Work, and Challenges

This section describes a typical wafer manufacturing assembly line and discusses the data characteristics, related work, and challenges for causal structure learning.

4.2.1 Wafer Manufacturing

As depicted in Figure 4.1, a wafer in an assembly line begins with an Aluminum-Titanium-Carbon (AlTiC) substrate. It goes through many stages, each involving multiple processing and measuring steps. Processing consists of several categories: photo-lithography, etching, deposition, plate, lift-off, and chemical mechanical polishing. At each processing step, the material is added to unfinished parts using a specific tool, where 24 tools are available for different tasks, each of which monitors the processing procedure with hundreds of sensors. These processing steps will repeat along the assembly line several times to enhance the product's quality. After each processing step, a measuring step measures the product quality using metrology tools to check for abnormalities. Each measuring step takes ten measurements to inspect different aspects of the product. Finally, the assembly line sends wafers to a final test station, where they are accepted if meeting the quality criteria and rejected otherwise.

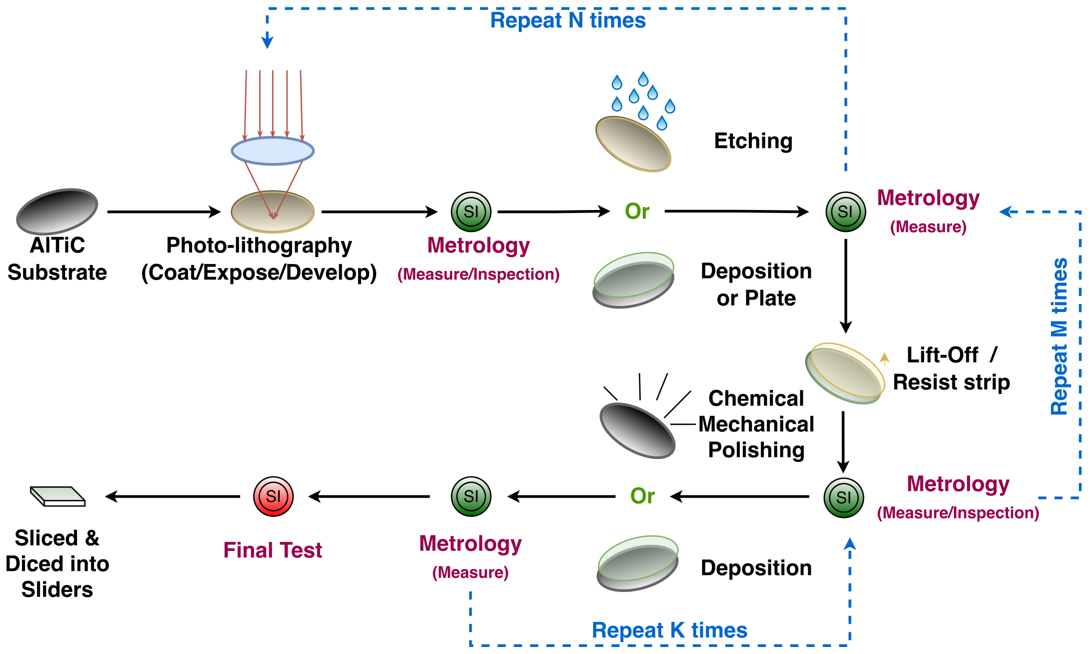


Figure 4.1: Wafer Manufacturing Process.

4.2.2 Data Characteristics, Related Work, and Challenges

The data in our case study came from a wafer assembly line in Seagate Technology and was collected from January 1, 2019, to October 11, 2021. The data exhibit the following characteristics, posing several challenges in learning the causal structures:

1. *High dimensionality*: The assembly line consists of more than 500 steps, each involving processing or measuring. Each processing step measures hundreds of sensor values, whereas each measuring step takes ten types of measurements. To make the learned causal structure useful for error tracing, we need a fine granularity and establish causal relations that capture which abnormal event or sensor at a particular step is the cause of an abnormality at another step. However, such a fine granularity requires the dataset used in causal discovery algorithms to contain all the variables from all steps, resulting in tens of thou-

sands of variables. Existing modifications targeting high-dimensional settings, such as FGES-MB (Ramsey et al., 2017) and ARGES (Nandy et al., 2018), can handle thousands of variables or millions of Gaussian variables with sparse graph structures. However, they are unsuitable for our case where the data is mixed and the dimension is much greater than the thousands level.

2. *Multimodality*: Two data sources come from measurement and sensor data. The measurement data record ten types of abnormalities at each measuring step, where each binary variable indicates whether a particular type of abnormality occurs. The sensor data include 24 sub-datasets, each corresponding to the sensor values when a wafer passes through a tool. The sensor data are continuous, whereas the measurement data are binary.

   For such a mixed data type, methods designed for a single type, such as PC (Spirtes et al., 2000) and GES (Chickering, 2002), are not applicable. Existing constraint-based mixed causal discovery methods, such as Copula-PC (Cui et al., 2016), symmetric conditional independence tests (Tsagris et al., 2018), CausalMGM (Sedgewick et al., 2019), and KAPC (Handhayani and Cussens, 2020), do not apply to high-dimensional situations due to their high computational complexity. Score-based methods, such as the Conditional Gaussian (CG) score and the Mixed Variable Polynomial (MVP) score (Andrews et al., 2018), require a large sample size in high dimensions and lack computational efficiency. Andrews et al. (2019) later proposed the degenerate Gaussian (DG) score, which has been shown to be consistent and efficient in high-dimensional settings. However, this approach must be more flexible in incorporating constraints and is hence unsuitable.

3. *Imbalance*: In a well-functioning assembly line, abnormalities rarely occur, which means that the binary variables in the measurement data are imbal-

anced. When applied to imbalanced data, many learning algorithms are biased towards the majority group (Krawczyk, 2016), which makes the estimated effects of binary variables too weak to be significant. In causal structure learning, this implies that the learned structure will mainly capture the relations among the continuous variables and miss a significant proportion of those regarding binary variables, as shown in Table 4.2. Moreover, since multiple imbalanced binary variables exist, common downsampling or oversampling techniques for classification (Chawla et al., 2002) are not applicable. Downsampling or oversampling one variable might lead to no information on the other variables. This problem has been pointed out in fields like atmosphere (Barnes et al., 2019) and Earth systems (Runge et al., 2019), but effective methods have yet to be proposed in the literature as far as we know.

4. *Missingness*. Ideally, wafers move along the assembly line following a prespecified order, but in practice, they often skip some steps based on their states. Wafers skip different steps from each other irregularly, which leads to an unstructured missing pattern.

   Following Little and Rubin (2019), there are three missing types: missing completely at random (MCAR), missing at random (MAR), and missing not at random (MNAR). In the MCAR case, two straightforward approaches apply to missing data: simply deleting observations with at least one missing value or performing imputation on the missing values. The former may lead to downgraded performance with a reduced sample size due to reduced statistical power (Städler and Bühlmann, 2012; Tu et al., 2019), whereas the latter may introduce biases in modeling the data distribution (Kyono et al., 2021).

   Researchers have made much progress in handling missing data in causal discovery in recent years. Sokolova et al. (2017) proposed a method to handle mixed and MAR data simultaneously. However, they assumed the continuous variables follow a non-paranormal distribution, and their estimation was based on the computationally expensive EM algorithm. Tu et al. (2019) proposed Missing Value PC (MVPC) that has been shown to be asymptotically correct even on data that are MNAR, but just like the PC algorithm, it cannot handle high-dimensional and mixed data. Regarding the MCAR case, Gao et al. (2022) proposed MissDAG upon additive noise models. However, it only applies to continuous data, and its estimation involves Monte Carlo EM, which can be time-consuming in high dimensions.

The wafer data in our case study could contain variables of all three types, and due to the enormous problem scale, it is hard to identify the missing type for each variable. None of the existing methods suits our needs perfectly. If we make a compromise regarding the missing type and consider all the missing variables are MCAR and try the naive approach, which simply deletes the samples with at least one missing value, the challenge remains in that the merged data contains too many variables which might be missing, and after the deletion, no observations are left.

5. *Temporal & Domain Constraints.* The production process imposes a natural temporal order onto measurements, suggesting the so-called causal order of occurrences of two events. Moreover, each step measures its data values simultaneously, making it sensible to assume the absence of contemporaneous effects. Namely, the variables at the same step are not causally related. In addition, certain causal relationships are not plausible based on domain knowledge. For example, two far-away nodes have no linkage; see Appendix D.2 for details. Properly leveraging these temporal and domain constraints can reduce the causal structure space complexity and improve the quality of structure

learning. Failing to do so would lead to factual errors in the learned structure. However, incorporating constraints into some advanced algorithms (Andrews et al., 2019) is not straightforward.

In what follows, we propose tailored methods to strike a balance in addressing these challenges simultaneously.

4.3 Preliminaries

4.3.1 Directed Graphical Causal Models

Here, we use directed graphical causal models (DGCMs) to represent causal relations. A DGCM consists of three main components: (1) a set of variables $\mathbf{X}$; (2) a set of directed edges E in a graph G; (3) a joint probability distribution characterized by

$$P(\mathbf{X}) = \prod_{i=1}^{p} P\left(X_i \mid \mathrm{Pa}(G, X_i)\right), \tag{4.1}$$

where $\mathrm{Pa}(G, X_i)$ is the set of all parents of X_i in the graph.

To interpret causal relations, we require that X_i is a direct cause of X_j if $X_i \to X_j \in E$, that is, an intervention on X_i changes the distribution of X_j when keeping the other variables fixed (Glymour et al., 2019). For a directed acyclic graph (DAG), learning causal relations from observational data requires three assumptions (Spirtes et al., 2000; Pearl, 2009; Scutari and Denis, 2021).

1. *Causal Markov.* Each variable $X_i \in \mathbf{X}$ is conditionally independent of its non-descendants given its parents.

2. *Causal Faithfulness.* There must exist a DAG faithful to the joint probability distribution P, meaning that only the dependencies arising from d-separation (Pearl, 2009) in the DAG can appear in P.

3. *Causal Sufficiency.* There cannot be any unmeasured variables acting as confounding factors.

4.3.2 Causal Discovery

Causal discovery works on learning the causal structure out of observational data, and there are generally three types of approaches: constraint-based, score-based, and functional causal model (FCM) based. Constraint-based algorithms, such as PC (Spirtes et al., 2000) and FCI (Spirtes et al., 2000), estimate the graph skeleton by conditional independence tests and orient the edges following some rules. Score-based algorithms use heuristics to learn the causal structure by maximizing a score function that quantifies the goodness of fit of the graphical network, for example, BIC (Maxwell Chickering and Heckerman, 1997) and BDe(u) (Heckerman et al., 1995). Finally, FCM-based algorithms, such as LiNGAM (Shimizu et al., 2006), parameterize a functional causal model in the form of $X_i = f(\mathrm{Pa}(G, X_i), e_i)$ and define a graph from the estimated model.

4.4 Methodology

As discussed in Section 4.2.2, no existing approaches can handle the five challenges at the same time. Therefore, we need to make some compromises and try to balance these aspects. In the design of our proposed method, scalability and the capability to handle missingness is of top priority, followed by constraint incorporation and mixed data modeling.

In the rest of the section, we first discuss data preprocessing and how to incorporate the constraints. Then, we introduce three constrained causal structure learning methods for mixed data and describe the hierarchical ensemble modeling strategy. Finally, we describe three model evaluation metrics and the overall pipeline in production.

4.4.1 Data Preprocessing

Apart from some primary data preprocessing described in Appendix D.1, we apply principal component analysis (PCA) to alleviate high-dimensionality. At each processing step, we perform PCA to generate m_{step} principal components to reduce the sensor data dimension. These principal components would then be the sensor-related nodes in the learned graph. To trace back to a specific sensor, we keep a record of the principal components' loading matrix to identify which sensors are the most relevant given a principal component associated with an abnormality's cause[1].

4.4.2 Incorporating Constraints

To incorporate the temporal and domain constraints, we transform them into implausible causal relations. For each step on the assembly line, set a unique time stamp[2] associated with the sensor and measurement variables to represent the manufacturing order on the assembly line. Let $\mathbf{X}_t$ represent all the variables at time t. The temporal constraints then imply that the variables at time $t+j$ cannot be the causes of those at time t, $\forall j \geq 0$, namely, $\{X \rightarrow X' \mid \forall X \in \mathbf{X}_t, \forall X' \in \mathbf{X}_{t'}, t \geq t' \geq 1\}$ are implausible. Similarly, we derive the implausible links based on the domain knowledge in Appendix D.2. Different algorithms use these implausible relations differently, as explained subsequently.

[1]Recall that the loading matrix of PCA contains the weights for each original variable when calculating the principal components, so we can check the absolute weight values in the loading matrix to identify which sensors are contributing the most to the principal component.

[2]Note that the time stamp here is different from that in time-series causal discovery in that each time stamp here represents the manufacturing order of different steps and includes different sets of sensors or measurements whereas time-series causal discovery focuses on the same set of variables that change over time.

4.4.3 Constrained Causal Structure Learning on Mixed Data

To treat the issue of mixed data types, we propose three causal structure learning methods: Discretization-based learning, Regression-based learning, and Constrained Kernel Alignment PC (CKAPC). Specifically, the first method discretizes all continuous variables and uses existing methods to learn a discrete causal network. The second fits node-wise regularized generalized linear regressions and then combines the selected variables to form the global causal structure. Finally, the third applies the idea of kernel alignment to calculate a pseudo-correlation matrix on the mixed data and then feeds it into the PC algorithm (Spirtes et al., 2000) to learn the causal structure.

Although more advanced mixed data causal discovery methods exist, they are either inflexible with constraints incorporation (Andrews et al., 2019) or computationally expensive in high-dimensions (Cui et al., 2016; Andrews et al., 2018). The three proposed methods are sub-optimal but allow for convenient constraint insertion and represent three schemes for distilling constraints. Examining all three would give us a more comprehensive idea of the suitable modeling choice.

Discretization-based Learning

Given mixed data, one straightforward idea is to discretize all continuous variables into factors and then learn a discrete DAG. Discretization allows us to employ existing methods and improves computational speed, with the price of losing potentially important information. Commonly used discretization methods include quantile discretization, interval discretization, and information-preserving discretization (Scutari and Denis, 2021).

After discretizing the data, we apply the existing methods[3] to learn a discrete

[3]See Chapter 6 of Scutari and Denis (2021) for a comprehensive review.

network. To ensure the constraints are satisfied, we input the list of implausible relations as the blacklist argument in the learning algorithms (Scutari and Denis, 2021).

Regression-based Learning

Recall that regression usually cannot determine the causes and effects due to the bi-directionality of correlation. For example, if a regression model $Y \sim X$ is significant on X, then the causal relation can be either $X \to Y$ or $Y \to X$ given that there are no confounders. Traditional FCM-based methods, such as LiNGAM (Shimizu et al., 2006), need to assume the non-Gaussian noise and apply independent component analysis (ICA) (Hyvarinen, 1999) to find the causal order. However, with prior knowledge of $t_X < t_Y$, we can infer that only $X \to Y$ is plausible. Therefore, temporal constraints open the door to learning causal relations via regressions.

As in Sedgewick et al. (2019), we assume the data follow a Mixed Graphical Model (MGM) (Lee and Hastie, 2015), where Gaussian linear and logistic regression specifies the conditional distributions. Given the conditional distributions, we identify the conditional independence directly from regression coefficients, with nonzero indicating conditional dependence. For each node X_i, we solve a regularized conditional log-likelihood problem subjective to constraints:

$$\begin{aligned} &\min_{\theta} \ell\left(\theta; X_i \mid \mathbf{X}_{\setminus i}\right) + \lambda \cdot Penalty(\theta), \\ &\text{s.t. } \theta_j = 0 \text{ if } X_j \to X_i \text{ is implausible by constraints,} \end{aligned} \tag{4.2}$$

where $\ell(\cdot)$ is the log-likelihood function, θ_j is the coefficient for variable $X_j \in \mathbf{X}_{\setminus i}$, $Penalty(\cdot)$ is the penalty function that induces sparsity[4], and λ is the tuning param-

[4]Common choices include LASSO (Tibshirani, 1996), SCAD (Fan and Li, 2001), MCP (Zhang, 2010), and TLP (Shen et al., 2013).

eter for the penalty[5]. Then, we estimate the parent set as $\widehat{Pa}(X_i) = \left\{X_j : \hat{\theta}_j \neq 0\right\}$. After fitting the regression models for all nodes, we obtain the overall structure as $\widehat{E} = \cup_{X_i \in \mathbf{x}} \cup_{X_j \in \widehat{Pa}(X_i)} \{(X_j \to X_i)\}$.

Constrained Kernel Alignment PC (CKAPC)

We adopt the KAPC method of Handhayani and Cussens (2020). First, we apply the RBF kernel for continuous variables and the Categorical Kernel (Belanche Muñoz and Villegas, 2013) for binary variables while transforming each variable into a single Gaussian variable in the feature space. Then we apply the idea of kernel alignment to build a pseudo-correlation matrix. The original KAPC method considers no constraints, whereas CKAPC reinforces the constraints by modifying the PC algorithm.

Starting with a complete graph, we remove the contemporaneous edges and then orient the edges by temporal constraints, after which we remove implausible ones from domain constraints. On top of that, we then perform conditional independence tests using the pseudo-correlation matrix to remove more edges and get the ultimate structure. Since all edges are oriented temporally, we do not have to use the orientation rules as in the traditional PC algorithm.

4.4.4 Hierarchical Ensemble Modeling

As mentioned in Section 4.2.2, although existing causal discovery methods for missing data can handle the MCAR (Sokolova et al., 2017; Gao et al., 2022) and even the MNAR case (Tu et al., 2019), they are usually computationally expensive and do not fit high-dimensional and mixed data. As a compromise, we assume all missing variables are MCAR and employ a straightforward approach that deletes samples with at least one missing value. However, suppose we try to learn the causal structure in

[5]We tune λ using cross-validation in each node-wise regression model.

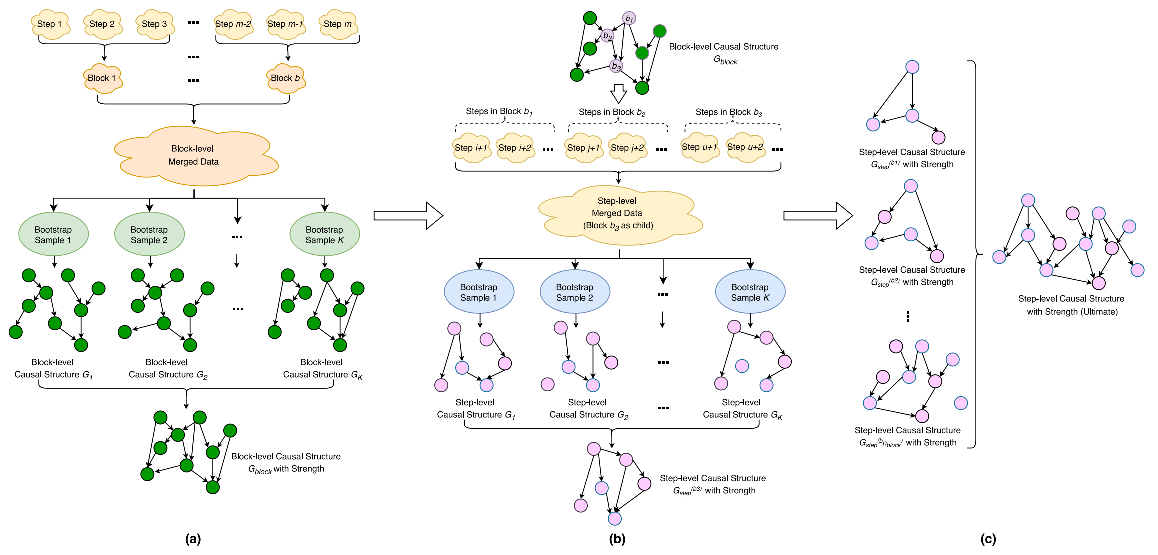

Step 1
Step 2
Step 3
Step m-2
Step m-1
Step m
Block 1
Block b
Block-level Merged Data
Bootstrap Sample 1
Bootstrap Sample 2
Bootstrap Sample K
Block-level Causal Structure G_1
Block-level Causal Structure G_2
Block-level Causal Structure G_K
Block-level Causal Structure G_{block} with Strength
(a)
Block-level Causal Structure G_{block}
Steps in Block b_1
Steps in Block b_2
Steps in Block b_3
Step i+1
Step i+2
Step j+1
Step j+2
Step u+1
Step u+2
Step-level Merged Data (Block b_3 as child)
Bootstrap Sample 1
Bootstrap Sample 2
Bootstrap Sample K
Step-level Causal Structure G_1
Step-level Causal Structure G_2
Step-level Causal Structure G_K
Step-level Causal Structure $G_{step}^{(b3)}$ with Strength
(b)
Step-level Causal Structure $G_{step}^{(b1)}$ with Strength
Step-level Causal Structure $G_{step}^{(b2)}$ with Strength
Step-level Causal Structure $G_{step}^{(b_{n_{block}})}$ with Strength
Step-level Causal Structure with Strength (Ultimate)
(c)

Figure 4.2: Hierarchical ensemble modeling. (**a**) for block-level data processing and structure learning, (**b**) for step-level data processing and structure learning, and (**c**) for union aggregation of step-level structures.

one run and use the overall merged data, which contains all the variables from all steps. Then, after applying the missing deletion strategy, we are likely to end up with few samples[6]. In response, we propose a hierarchical modeling strategy, which learns the causal structure at two granularity levels. In addition, we fuse in the ensemble idea similar to that in random forest (Breiman, 2001) to alleviate the imbalance problem and to provide confidence measures for the learned relations. The modeling framework is shown in Figure 4.2.

Hierarchical Modeling

There are three main steps in hierarchical modeling: block-level learning, step-level learning, and aggregation. In order to perform this modeling strategy, domain knowledge is required in terms of which steps along the assembly line can be clustered into a block. The block-level learning will capture which two blocks are causally

[6]In our case study, we get no samples at all, making it impossible to learn the structure.

related, whereas step-level learning is finer-grained and captures which two sensor/measurement variables are causally related. Moreover, once we learn the block-level structure, we can perform the step-level learning in parallel and then union all step-level results to get the final structure.

1. *Block-level Data Processing and Structure Learning.* After discussing with expert engineers and establishing n_{block} blocks, for each block, we aggregate the measurement data such that as long as a certain type of abnormality occurs inside this block, no matter in which step, we label the corresponding abnormality type indicator to be 1. And the sensor data are aggregated by PCA with m_{block} principal components. The processed data for each block contains ten measurement variables and m_{block} sensor-data variables. Next, we merge the n_{block} block-level data sets into one big data set, with rows representing wafers and columns representing measurement or sensor variables from blocks. This data set has enough records for learning in that although few wafers go through all steps, many wafers pass through all blocks. Based on this data, we apply the proposed constrained mixed-data causal structure learning methods to learn a causal graph where each node is a measurement or sensor-related variable in a certain block. Finally, we map this graph to a block-level graph G_{block} where each node represents a block.

2. *Step-level Data Processing and Structure Learning.* For the jth block b_j, $1 \leq j \leq n_{\text{block}}$, we pick out its parent set in G_{block} and denote it as $Pa(G_{\text{block}}, b_j)$. We merge the data from all steps inside blocks $\{b_j\} \cup Pa(G_{\text{block}}, b_j)$. Since we are merging a subset of steps, we get much more observation records than merging across all steps. Based on the step-level merged data, we then learn a step-level causal structure $G_{\text{step}}^{(b_j)}$, where each node represents a measurement variable or a sensor-related variable from some step.

3. *Combination of Step-level Causal Structures.* In the end, we combine all the learned step-level causal structures and obtain the ultimate causal structure $\bigcup_{1\leq j\leq n_{\text{block}}} G_{\text{step}}^{(b_j)}$.

This scheme alleviates both the missingness and the high-dimensional problem. Instead of merging all steps and getting a few observations in high-dimension, we only merge the steps inside a subset of blocks, which contains much fewer steps and gives a decent number of records after missing deletion and a lower dimension in the merged data. Moreover, the hierarchical scheme grants us two levels of causal structures, providing the engineers with more insights. Meanwhile, it is worth noting that the quality of this scheme highly depends on the block division given by the engineers, so a thorough conversation with the engineers is crucial.

Ensemble Modeling

In manufacturing, including extra relations costs additional engineers' validation time, whereas missing true relations leads to false root cause diagnosis, which is much more costly. Also, missing block-level relations in hierarchical modeling leads to accumulative errors in subsequent step-level modeling. Therefore, we prefer conservative estimates and want to control the false negative rate. We employ stability selection (Meinshausen and Bühlmann, 2006) and bagging (Breiman, 1996) to construct an ensemble model for both the block-level and the step-level structure learning. First, we resample the original data with replacement to generate K new data sets, each of the original data sample size. Then, we learn a causal structure G_k on the kth resampled data for $\forall k \in \{1, 2, \cdots, K\}$. Finally, different from the threshold aggregation in Meinshausen and Bühlmann (2006) and mean aggregation in Breiman (1996), we union these K learned structures and obtain $\cup_{k=1}^{K} G_k$ as the ultimate causal structure. The temporal constraint enforced on the structures guarantees the directions of causal relations.

There are three benefits of ensemble modeling. First, it helps alleviate the bias towards the majority group and the weak signal problem due to imbalanced data. With ensemble modeling, we have many resampled data, some of which might magnify the weak signals of certain imbalanced variables. Every single model acts as a weak learner to capture the partial information, granting the ensemble model more capability to detect signals. Second, the estimate is stabler with the union aggregation, which is crucial for engineers' validation and technicians' usage. Third, it allows us to quantify the strength of the learned relations. As in Equation (4.3), we define the strength of a causal relation e in $\cup_{k=1}^{K} G_k$ as the proportion it appears in the K learned structures. Such a measure not only quantifies credibility but also helps with validation. Given a large number of learned causal relations to be fully validated, engineers can start by checking the ones with strengths above a threshold and provide feedback to the modelers timely. In this way, the pipeline loop runs at a faster pace. Also, with the reduced workload, the engineers would be more willing to get involved, a factor we must consider in a real-life production environment.

$$s_e = \frac{\sum_{k=1}^{K} \mathbf{1}\left(e \in G_k\right)}{K}, \quad \forall e \in \cup_{k=1}^{K} G_k. \tag{4.3}$$

4.4.5 Evaluation

We propose three metrics to evaluate the results, collectively assessing a method's performance.

1. *Engineers' Validation.* Given the learned causal relations and associated strengths, domain engineers validate the causal relations with strengths higher than a pre-specified threshold.

2. *Comparison against Known Knowledge.* Given some known relations on which

two steps are causally related, we validate the results by transforming the learned relations to step-wise relations (if $\exists X \in \text{step}_i, Y \in \text{step}_j$ s.t. $X \rightarrow Y$, then the corresponding step-wise relation is $\text{step}_i \rightarrow \text{step}_j$).

3. *Conditional Independence Tests.* Given a learned network, we select a few nodes of interest and test if the conditional independence relationship given by d-separation (Pearl, 2009) is the same as that by conditional independence tests on the data. Specifically, if S d-separates X from Y according to the learned graph, implying $X \perp\!\!\!\perp Y|S$, but the conditional independence tests tell us that $X \not\perp\!\!\!\perp Y|S$, then this relationship is not trustworthy. We consider three conditional independence tests for the mixed data, including Pearson X^2 test, the symmetric conditional independence test (Tsagris et al., 2018), and the CODEC measure (Azadkia and Chatterjee, 2021).

4.4.6 Modeling Pipeline in Production

The overall modeling pipeline is a positive feedback loop, as shown in Figure 4.3. First, we query data from the databases and consult expert engineers to establish the domain constraints. Second, we apply hierarchical ensemble modeling to learn the causal relations among the variables along the assembly line. Third, we validate the learned results through comparisons with the known causal relations and via engineers' validation. When the engineers determine an identified relationship to be definite, we add it to the inclusion (if true) or the exclusion (if false) databases. Otherwise, we take no action. Afterward, the engineers update the constraints based on these included and excluded relations.

This approach applies to online and offline data. For a streaming pipeline, we dynamically update the database and causal structures. Otherwise, we circle through the pipeline for several rounds by updating the constraints and rerunning the structure

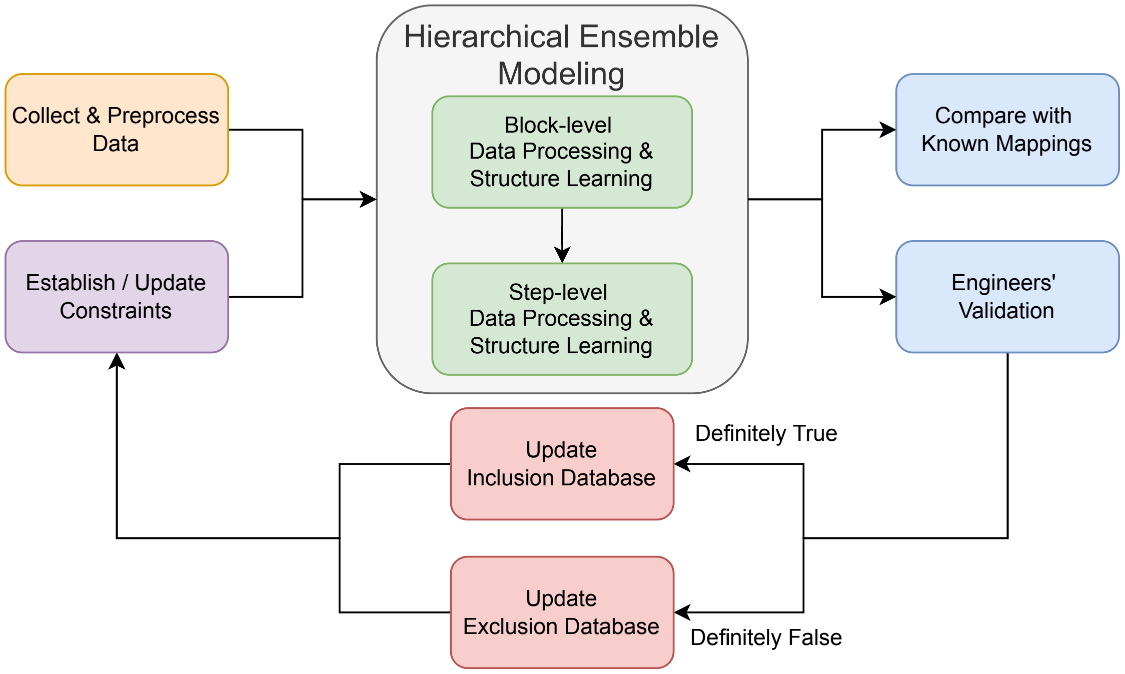

Hierarchical Ensemble Modeling
Collect & Preprocess Data
Establish / Update Constraints
Block-level Data Processing & Structure Learning
Step-level Data Processing & Structure Learning
Compare with Known Mappings
Engineers' Validation
Update Inclusion Database
Update Exclusion Database
Definitely True
Definitely False

Figure 4.3: Modeling Pipeline in Production.

learning model. As a result, we keep improving the accuracy of the learned structure. The inclusion and exclusion databases also serve as a knowledge base for technicians to trace the root causes of failure events. In some situations, the technicians can check the databases to resolve an issue without consulting an expert engineer.

4.5 Experiments

We perform simulation studies and real data applications to verify the effectiveness of the proposed method.

4.5.1 Simulation Studies

Since existing methods are either not scalable or not flexible with constraints incorporation, we do not make comparisons with them in simulations. Instead, we focus on investigating the operating characteristics of the proposed methods under different situations to provide a better understanding and guidance to practitioners.

Effects of Mixed Data Structure Learning Methods

We first examine the proposed constrained mixed-data structure learning methods. The simulation mimics the assembly line subject to temporal constraints. We vary the number of observations $n \in \{600, 6000\}$, the number of variables $p \in \{20, 50, 100, 600\}$, and the number of groups (to mimic steps) $k \in \{2, 5, 10, 60\}$, while fixing the group size to be 10, the vertex degree to be 4, and the proportion of continuous variables to be 0.2[7]. We first generate graphs with the constraints enforced and then generate the data following the MGM framework (Lee and Hastie, 2015). All simulations run for 30 repetitions with different seeds and are implemented on a Linux server with an Intel(R) Xeon(R) CPU E5-2686 v4 @ 2.30GHz and 124GB RAM.

We compare six methods, including three discretization-based methods: *tabu-bic* (Tabu search (Glover, 1989, 1990) with BIC score), *tabu-bde* (Tabu search with BDe score), and *pc.stable* (PC-Stable algorithm (Colombo et al., 2014)); two regression-based methods: *lasso-min* (Lasso-penalized regressions (Tibshirani, 1996) with $\lambda_{\min}$ in cross-validation) and *lasso-1se* (Lasso-penalized regressions with λ_{1se} in cross-validation); and one CKAPA method ($\sigma = 0.01$ for the RBF kernel; $\theta = 1$ for the Categorical kernel). Four metrics are employed to quantify the model performance: adjacency precision (AP), adjacency recall (AR), Structural Hamming Distance (SHD)[8], and elapsed time.

[7]We pick these numbers to mimic the real data in our case study.

[8]SHD measures the number of edge insertions, deletions or flips needed to transform one graph

We report the mean performance and their standard deviations (in parentheses) in Table 4.1. Discretization-based learning performs the best in AP, while regression-based learning is usually the best in AR, which we anticipate since Lasso tends to over-select variables. Regarding SHD, *tabu-bic* and *pc.stable* are the top two across all settings. Concerning runtime, *tabu-bic* is mostly the fastest, especially in high dimensions, whereas CKAPC is too slow to complete all tasks except Setting 5.

Effects of Ensemble on Data Imbalance

We also investigate how the ensemble strategy alleviates the weak signal problem due to data imbalance. We simulate the data as before, with $n = 3000, p = 200, k = 20$, and an additional hyper-parameter controlling the imbalance ratio. All simulations run ten repetitions with different seeds and are implemented in parallel on 20 CPU cores. We use *tabu-bic* as the base model and set the number of weak learners $K = 1000$ in ensemble models. We compare two model types: single models and ensemble models with a strength threshold $\alpha \in \{0, 0.05, 0.1, \cdots, 0.95, 1\}$. The focus is to reduce the false negative rate, so we consider different types of AR while controlling the overall AP.

Table 4.2 shows the mean performance and their standard deviations with $\alpha \in \{0, 0.05, 0.1, 0.15, 0.2\}$[9] under six imbalanced scenarios. We can see that data imbalance impacts the AR, particularly those relating to discrete nodes. If we can tolerate a low AP, then *ensem_0* (ensemble model with no threshold) greatly helps identify the edges the base model easily misses. If, however, we want to control the AP at an acceptable level, then $\alpha \in [0.05, 0.2]$ would be a good choice, and a smaller α gives higher ARs but a lower AP.

to another (Tsamardinos et al., 2006).

[9]For clarity, we do not show all the results here. Refer to Yang (2023) for full results with other thresholds and other types of AP.

Setting	Method	AP ↑ [a]	AR ↑	SHD ↓	Elapsed Time [b]↓
Setting 1 *(n=600 p=100 k=10)*	tabu-bic	0.863 (0.037)	0.497 (0.036)	**88.6 (10.988)**	**0.693 (0.063)**
	tabu-bde	0.848 (0.049)	0.478 (0.032)	92.567 (12.016)	0.709 (0.072)
	pc.stable	**0.882 (0.041)**	0.475 (0.038)	89.867 (10.67)	5.087 (0.523)
	lasso-min	0.26 (0.02)	0.735 (0.039)	360.467 (47.963)	15.064 (2.428)
	lasso-1se	0.664 (0.056)	**0.601 (0.042)**	107.733 (14.249)	15.985 (4.25)
Setting 2 *(n=6000 p=100 k=10)*	tabu-bic	**0.967 (0.018)**	0.742 (0.038)	**43.3 (7.452)**	**2.843 (0.291)**
	tabu-bde	0.963 (0.021)	0.73 (0.04)	45.333 (7.941)	2.901 (0.267)
	pc.stable	0.929 (0.025)	0.759 (0.038)	45.833 (8.91)	7.207 (0.947)
	lasso-min	0.259 (0.021)	**0.911 (0.022)**	412.167 (44.772)	65.127 (5.172)
	lasso-1se	0.878 (0.056)	0.814 (0.035)	45.767 (10.251)	65.065 (5.106)
Setting 3 *(n=600 p=600 k=60)*	tabu-bic	**0.702 (0.016)**	0.497 (0.012)	799.133 (25.704)	40.766 (4.482)
	tabu-bde	0.686 (0.022)	0.478 (0.011)	828.8 (28.602)	41.905 (3.814)
	pc.stable	0.796 (0.019)	0.435 (0.017)	**756.767 (30.514)**	5449.623 (142.001)
	lasso-min	0.171 (0.006)	**0.695 (0.014)**	4110.5 (178.198)	1888.124 (187.423)
	lasso-1se	0.48 (0.027)	0.599 (0.015)	1177.133 (79.856)	1904.291 (170.195)
Setting 4 *(n=6000 p=600 k=60*	tabu-bic	**0.908 (0.011)**	0.746 (0.011)	**368.567 (15.174)**	**117.23 (13.325)**
	tabu-bde	0.9 (0.014)	0.73 (0.01)	392.267 (17.066)	119.978 (13.454)
	pc.stable	0.901 (0.011)	0.716 (0.014)	405.667 (22.202)	5792.206 (203.995)
	lasso-min	0.184 (0.008)	**0.898 (0.011)**	4580.933 (247.047)	4804.765 (587.206)
	lasso-1se	0.782 (0.033)	0.826 (0.013)	453.433 (54.219)	4772.73 (565.064)
Setting 5 *(n=6000 p=20 k=2)*	tabu-bic	**0.997 (0.014)**	0.748 (0.135)	2.9 (1.826)	0.363 (0.041)
	tabu-bde	0.99 (0.032)	0.748 (0.144)	2.967 (2.025)	0.35 (0.041)
	pc.stable	0.972 (0.047)	**0.797 (0.098)**	**2.633 (1.671)**	**0.329 (0.029)**
	lasso-min	0.408 (0.224)	0.762 (0.355)	13.2 (9.264)	2.916 (1.354)
	lasso-1se	0.799 (0.37)	0.649 (0.315)	3.533 (1.795)	2.898 (1.373)
	ckapc [c]	0.228 (0.099)	0.769 (0.19)	36.833 (21.001)	277.488 (345.531)
Setting 6 *(n=6000 p=50 k=5)*	tabu-bic	**0.984 (0.023)**	0.745 (0.068)	**15.533 (3.875)**	**0.896 (0.104)**
	tabu-bde	0.979 (0.028)	0.732 (0.067)	16.5 (4.125)	0.915 (0.116)
	pc.stable	0.936 (0.044)	0.782 (0.055)	15.833 (4.684)	1.083 (0.159)
	lasso-min	0.316 (0.04)	**0.916 (0.049)**	122.3 (22.426)	15.409 (1.862)
	lasso-1se	0.91 (0.056)	0.807 (0.06)	15.8 (3.8)	15.423 (1.636)

[a] ↑ means the higher the better, ↓ means the lower the better.
[b] In seconds.
[c] Settings 1 - 4 and Setting 6 have no *ckapc* results as CKAPC is too slow to be finished.

Table 4.1: Simulation Results on Constrained Mixed Data Causal Structure Learning Methods.

Imbalance Ratio [a]	Model	Overall AP	Overall AR	To Discrete AR [b]	From Discrete AR [b]	From Discrete To Discrete AR [b]	To Continuous AR [c]	From Continuous AR [c]	From Continuous To Continuous AR [c]
0.5051 (0.0246)	ensemble_0 [d]	0.0594 (0.0044)	0.8805 (0.008)	0.9048 (0.0164)	0.8614 (0.0207)	0.8961 (0.0261)	0.8583 (0.0131)	0.9017 (0.0169)	0.8941 (0.0314)
	ensemble_0.05	0.5178 (0.0242)	0.7909 (0.0147)	0.7903 (0.0217)	0.7364 (0.0344)	0.7308 (0.0516)	0.792 (0.0185)	0.8493 (0.028)	0.8492 (0.0354)
	ensemble_0.1	0.7183 (0.0199)	0.7741 (0.0182)	0.7689 (0.0261)	0.7131 (0.0363)	0.7017 (0.0533)	0.7795 (0.02)	0.8397 (0.0283)	0.843 (0.035)
	ensemble_0.15	0.8214 (0.0193)	0.7646 (0.0187)	0.757 (0.0242)	0.6985 (0.0364)	0.685 (0.052)	0.7718 (0.0225)	0.8355 (0.029)	0.8414 (0.033)
	ensemble_0.2	0.8729 (0.018)	0.7584 (0.0168)	0.7531 (0.0219)	0.6883 (0.0364)	0.6797 (0.0485)	0.764 (0.02)	0.8337 (0.0293)	0.8404 (0.0328)
	single	0.9379 (0.0124)	0.7056 (0.0206)	0.6965 (0.0276)	0.6229 (0.0399)	0.6127 (0.0488)	0.715 (0.0178)	0.7943 (0.024)	0.8078 (0.0273)
0.2524 (0.0099)	ensemble_0	0.0514 (0.0039)	0.8673 (0.013)	0.8866 (0.0199)	0.8502 (0.0236)	0.8839 (0.0337)	0.8494 (0.0139)	0.8856 (0.0265)	0.8871 (0.0277)
	ensemble_0.05	0.4794 (0.0267)	0.7432 (0.0272)	0.7084 (0.037)	0.6721 (0.0455)	0.6267 (0.0626)	0.7778 (0.0221)	0.8192 (0.0309)	0.8492 (0.0299)
	ensemble_0.1	0.696 (0.0209)	0.7148 (0.0322)	0.6648 (0.039)	0.6328 (0.0467)	0.5645 (0.0577)	0.764 (0.0261)	0.8034 (0.0352)	0.8397 (0.0252)
	ensemble_0.15	0.816 (0.0173)	0.7 (0.0299)	0.6427 (0.0367)	0.6152 (0.0454)	0.5388 (0.0566)	0.7561 (0.0227)	0.7915 (0.0312)	0.8344 (0.0243)
	ensemble_0.2	0.8789 (0.0168)	0.6881 (0.0316)	0.6201 (0.0368)	0.5985 (0.0441)	0.507 (0.0509)	0.7543 (0.0238)	0.7849 (0.0332)	0.8344 (0.0243)
	single	0.9359 (0.0125)	0.6347 (0.0273)	0.5503 (0.0274)	0.5356 (0.0426)	0.4294 (0.0546)	0.7164 (0.0244)	0.7418 (0.0257)	0.8111 (0.0325)
0.1005 (0.0034)	ensemble_0	0.0423 (0.0025)	0.8526 (0.0135)	0.8793 (0.0223)	0.81 (0.028)	0.8531 (0.0394)	0.8278 (0.0255)	0.8994 (0.0209)	0.8966 (0.0285)
	ensemble_0.05	0.3965 (0.0203)	0.6408 (0.0286)	0.5396 (0.0235)	0.5148 (0.0352)	0.3815 (0.0431)	0.7376 (0.0287)	0.7767 (0.0306)	0.8515 (0.0239)
	ensemble_0.1	0.611 (0.024)	0.6003 (0.038)	0.4741 (0.0362)	0.4626 (0.0499)	0.2984 (0.0566)	0.7218 (0.0358)	0.7498 (0.031)	0.8425 (0.0203)
	ensemble_0.15	0.7421 (0.0217)	0.5792 (0.0417)	0.4391 (0.0359)	0.4356 (0.0548)	0.2558 (0.0511)	0.7136 (0.038)	0.7353 (0.0318)	0.8383 (0.0209)
	ensemble_0.2	0.8124 (0.0222)	0.5614 (0.0413)	0.414 (0.0374)	0.4133 (0.0544)	0.2244 (0.0545)	0.7029 (0.0373)	0.7226 (0.0292)	0.8319 (0.0209)
	single	0.9002 (0.0215)	0.5136 (0.0262)	0.3482 (0.0204)	0.3599 (0.0371)	0.1641 (0.0334)	0.6725 (0.0333)	0.6808 (0.0224)	0.8187 (0.0264)
0.0503 (0.0018)	ensemble_0	0.0387 (0.0025)	0.8159 (0.0262)	0.8395 (0.0305)	0.7478 (0.0304)	0.7954 (0.0387)	0.7941 (0.0356)	0.89 (0.0311)	0.9006 (0.0259)
	ensemble_0.05	0.3438 (0.0192)	0.5534 (0.0265)	0.3922 (0.0249)	0.3859 (0.033)	0.1824 (0.0419)	0.7075 (0.0324)	0.7334 (0.0342)	0.8559 (0.0293)
	ensemble_0.1	0.5584 (0.0198)	0.5089 (0.0291)	0.324 (0.0251)	0.3371 (0.0319)	0.1136 (0.0337)	0.6849 (0.0352)	0.6943 (0.0237)	0.8428 (0.0221)
	ensemble_0.15	0.7016 (0.0193)	0.4872 (0.0316)	0.2911 (0.0263)	0.3135 (0.0346)	0.0857 (0.0231)	0.6742 (0.0334)	0.675 (0.0216)	0.8406 (0.0216)
	ensemble_0.2	0.7851 (0.0178)	0.4683 (0.0316)	0.2654 (0.022)	0.2932 (0.0322)	0.0646 (0.0173)	0.6622 (0.033)	0.6577 (0.0277)	0.8371 (0.0215)
	single	0.8665 (0.0293)	0.4231 (0.0225)	0.2188 (0.0273)	0.2476 (0.022)	0.0395 (0.0191)	0.6186 (0.0265)	0.613 (0.028)	0.8148 (0.0237)
0.0252 (0.001)	ensemble_0	0.0372 (0.0024)	0.7431 (0.0245)	0.7148 (0.0311)	0.6057 (0.0351)	0.5517 (0.0469)	0.7719 (0.0283)	0.8919 (0.018)	0.9033 (0.0272)
	ensemble_0.05	0.2904 (0.0237)	0.472 (0.031)	0.2825 (0.0172)	0.2794 (0.031)	0.0814 (0.0288)	0.6536 (0.0355)	0.6804 (0.0316)	0.8638 (0.0246)
	ensemble_0.1	0.4736 (0.0306)	0.432 (0.03)	0.2295 (0.0197)	0.2413 (0.0283)	0.0451 (0.0212)	0.6267 (0.034)	0.6379 (0.0287)	0.8505 (0.0224)
	ensemble_0.15	0.6231 (0.0394)	0.4091 (0.0317)	0.1974 (0.0206)	0.2235 (0.0276)	0.0301 (0.0143)	0.6124 (0.0361)	0.6095 (0.0348)	0.8434 (0.0275)
	ensemble_0.2	0.7246 (0.0363)	0.3949 (0.0326)	0.1743 (0.0253)	0.2131 (0.0248)	0.0176 (0.0138)	0.6067 (0.0338)	0.5907 (0.0373)	0.8408 (0.0251)
	single	0.8595 (0.0318)	0.3454 (0.0295)	0.1298 (0.0231)	0.163 (0.0204)	0.0058 (0.0079)	0.5512 (0.0326)	0.5417 (0.0334)	0.8188 (0.0243)
0.017 (0.0006)	ensemble_0	0.0383 (0.0024)	0.7029 (0.0203)	0.6448 (0.0319)	0.5328 (0.0262)	0.4346 (0.0555)	0.7583 (0.0207)	0.886 (0.0219)	0.9103 (0.023)
	ensemble_0.05	0.2687 (0.0203)	0.4386 (0.0318)	0.2413 (0.0346)	0.2445 (0.0275)	0.0665 (0.0328)	0.6285 (0.0325)	0.6478 (0.0288)	0.8692 (0.0275)
	ensemble_0.1	0.4565 (0.0329)	0.396 (0.0311)	0.1833 (0.029)	0.2087 (0.0252)	0.0388 (0.0175)	0.6008 (0.0384)	0.5976 (0.0286)	0.8578 (0.0282)
	ensemble_0.15	0.5971 (0.0383)	0.3729 (0.0273)	0.153 (0.0302)	0.1862 (0.0205)	0.0232 (0.0101)	0.5838 (0.0348)	0.5736 (0.0287)	0.8543 (0.0282)
	ensemble_0.2	0.7031 (0.0424)	0.3548 (0.0266)	0.1333 (0.0302)	0.1705 (0.0194)	0.0146 (0.0103)	0.567 (0.0336)	0.5535 (0.0333)	0.8446 (0.0266)
	single	0.8553 (0.0337)	0.3147 (0.0328)	0.0958 (0.024)	0.1356 (0.0167)	0.0069 (0.0077)	0.5239 (0.0339)	0.5072 (0.0451)	0.8208 (0.0231)

[a] The proportion of the minority in binary variables.

[b] The AR for edges pointing to discrete nodes / from discrete nodes / from discrete nodes to discrete nodes.

[c] The continuous counterparts have similar meanings as the discrete ones.

[d] *ensem*_α represents an ensemble model with threshold α, *single* represents a model without ensemble.

Table 4.2: Simulation Results on the Effect of Ensemble on Data Imbalance.

Effects of Hierarchical Modeling

We now investigate the effects of hierarchical modeling. The data are simulated similarly to before, except that we now impose a hierarchical structure by adding a few hyper-parameters: the number of blocks n_{block}, *missing_rate* which specifies the probability that a step is missing[10], *block_tightness*, which controls the relative closeness of the within-block versus between-block relations, and *max_step_distance*, which specifies the maximum steps apart for two connected nodes. We set $n = 2000, p = 600, k = 60, n_{\text{block}} = 10, \mathit{max\_step\_distance} = 30$, the imbalance ratio to be 0.1, $\mathit{missing\_rate} \in \{0, 0.02, 0.05, 0.1\}$, and $\mathit{block\_tightness} \in \{0.3, 0.5, 0.7, 0.9, 1.0\}$. We use *tabu-bic* as the base model, set the number of weak learners $\mathrm{K} = 1000$ in ensemble models, and perform hierarchical learning with strength threshold $\alpha = 0.2$ at both the block level and the step level. We examine both levels' precision, recall, SHD, and elapsed time. All simulations run ten repetitions with different seeds and are implemented in parallel on 20 CPU cores.

Table 4.3 shows the mean performance and their standard deviations. As the missing rate increases, the block-level recovery does not change much, whereas the step-level learning accuracy decreases significantly. When the missing rate is 0.02, close to the real data scenario, the step-level AP and AR are acceptable, implying the applicability to the real data. In addition, we note that high block tightness usually associates with low block-level recovery accuracy and high step-level AP[11]. In contrast, the step-level AR is pretty robust against the change in block tightness. As discussed in Section 4.4.4, we usually prefer to control the false negative rate in manufacturing. Hence, step-level AR is the most important out of all the metrics. Its

[10] To mimic the case study data, if a step is missing, then all variables in this step are missing. Moreover, we set the probability to be the same for all wafers and all steps in the simulation.

[11] This is as expected. For example, when block tightness is 1, only the initial node of each block is connected with other blocks, and hence the between-block relations are too weak to be well detected. Furthermore, once we get into the step level, the high value of block tightness implies smaller candidate parent node sets, and hence a sparser learned graph and a higher step-level AP.

robustness against block tightness is desirable because even if the steps are not well clustered, the final fine-grained AR will not be affected much.

4.5.2 Real Data Analysis

We followed the pipeline in Figure 4.3 and applied our proposed methods to the wafer data from Seagate Technology, which includes 24 processing tools, 2766 sensor types, ten measurement types, 125 stages, 592 steps, 68 blocks, and 66996 wafers. All ensemble models (with $K = 1000$) are run on 30 CPU cores in parallel on a Linux server with an Intel(R) Xeon(R) CPU E5-2686 v4 @ 2.30GHz and 124GB RAM.

We performed data preprocessing as described in Section 4.4.1, and set m_{step} to 4 for PCA on the sensor data. For block-level data processing, we set m_{block} to 4, creating merged block-level data with 8026 rows and 653 columns. We then applied the ensemble *tabu-bic* method, which has the shortest runtime and recovers the most known mappings from both simulations and analyses. The runtime for this phase was 2.25 hours. Finally, we discovered 1460 block-wise relations and identified 32 out of 46 known block-wise relationships.

After processing the data at the step level, we obtained 68 step-level datasets. We used the ensemble *tabu-bic* method to learn structures from these datasets and combined them to obtain the ultimate causal structure. The total time required for this process was approximately 6 hours. We discovered 9774 relationships among the measurement and sensor variables, corresponding to 1439 step-wise relationships. Moreover, out of 56 known step-wise relationships, we recovered 29 of them.

The engineers examined the causal relations with a strength of 1 (totaling 122), of which approximately 50% were found to be true, 20% were likely true but required more context to verify, and nearly 30% were likely false. The 50% definitely true relations were added to the inclusion database, while the 20% likely true and 30% likely false relations suggest that we have discovered unestablished relationships that

Missing Rate	Block Tightness [a]	Block-level AP ↑ [b]	Block-level AR ↑	Block-level SHD ↓	Step-level [c] AP ↑	Step-level AR ↑	Step-level SHD ↓	Elapsed Time [d]↓
0	0.3	**0.9358 (0.0476)**	**0.8109 (0.0556)**	**7.5 (2.6771)**	0.6248 (0.032)	0.5873 (0.0236)	660.8 (103.7827)	1.911 (0.4652)
	0.5	0.9124 (0.0499)	0.7472 (0.0462)	9.5 (2.0138)	0.65 (0.0278)	0.5898 (0.0256)	623.3 (79.0725)	2.229 (0.4902)
	0.7	0.8689 (0.0671)	0.7172 (0.1105)	10.8 (3.4897)	0.6854 (0.0228)	**0.5907 (0.0189)**	572.5 (40.4949)	0.8922 (0.3326)
	0.9	0.7585 (0.1001)	0.6938 (0.0912)	12.3 (3.3682)	0.7132 (0.0295)	0.582 (0.0223)	544.5 (43.2518)	**0.7701 (0.1766)**
	1	0.5047 (0.0645)	0.7098 (0.1787)	14.2 (3.1903)	**0.7256 (0.0245)**	0.5765 (0.0195)	**534.2 (38.8667)**	0.8945 (0.3986)
0.02	0.3	**0.9302 (0.052)**	**0.7755 (0.0535)**	**8.7 (2.2632)**	0.6377 (0.0297)	0.5225 (0.0275)	667.4 (95.6617)	1.5303 (0.3182)
	0.5	0.9062 (0.053)	0.7632 (0.0398)	9.3 (1.2517)	0.645 (0.0306)	0.5195 (0.0311)	654.5 (68.1929)	0.7342 (0.2213)
	0.7	0.8548 (0.0599)	0.7558 (0.0777)	10.4 (2.9515)	0.6814 (0.0213)	**0.5392 (0.022)**	599.4 (31.959)	1.1568 (0.307)
	0.9	0.7547 (0.0961)	0.691 (0.1114)	12.5 (2.9155)	0.7024 (0.0266)	0.5328 (0.0286)	**578.4 (43.5155)**	0.8314 (0.1795)
	1	0.536 (0.0871)	0.7274 (0.1206)	13.2 (2.8206)	**0.7074 (0.0332)**	0.516 (0.0329)	581.7 (44.7811)	**0.444 (0.13)**
0.05	0.3	**0.9373 (0.0446)**	0.7869 (0.0533)	8.1 (2.2336)	0.6118 (0.0284)	0.4254 (0.0353)	726.3 (91.6528)	0.55 (0.1023)
	0.5	0.9182 (0.0476)	**0.8045 (0.0582)**	**7.9 (2.0248)**	0.6245 (0.0196)	0.4227 (0.0371)	710.7 (66.7833)	1.2852 (0.2887)
	0.7	0.8969 (0.0498)	0.7472 (0.0864)	9.6 (2.4129)	0.6274 (0.0174)	0.4353 (0.0393)	691.8 (47.1659)	0.7723 (0.1202)
	0.9	0.7602 (0.0934)	0.6649 (0.1171)	12.6 (3.4705)	**0.6596 (0.0446)**	0.4432 (0.0331)	**655.9 (54.1489)**	0.3723 (0.0819)
	1	0.5313 (0.0765)	0.712 (0.1051)	13.2 (2.044)	0.6556 (0.0365)	**0.4434 (0.0369)**	657.3 (55.094)	**0.3517 (0.1105)**
0.1	0.3	**0.9327 (0.0509)**	0.7754 (0.0708)	8.6 (2.9515)	0.3785 (0.0645)	0.2455 (0.0312)	998.0 (98.1439)	0.3582 (0.0588)
	0.5	0.9243 (0.052)	**0.7904 (0.058)**	**8.1 (1.912)**	0.4198 (0.0317)	0.2502 (0.0168)	937.3 (80.0514)	0.4015 (0.0585)
	0.7	0.9064 (0.0787)	0.7566 (0.076)	9.0 (3.266)	0.4468 (0.0347)	0.2669 (0.034)	892.5 (68.6541)	0.461 (0.1066)
	0.9	0.7865 (0.0929)	0.6972 (0.0583)	11.5 (1.7795)	0.4739 (0.0451)	0.2967 (0.048)	860.2 (80.5161)	**0.2762 (0.0512)**
	1	0.5199 (0.0591)	0.7145 (0.1541)	13.5 (2.2236)	**0.4833 (0.0664)**	**0.3111 (0.0557)**	**850.2 (98.363)**	0.328 (0.1026)

[a] Probability that a non-initial node's parent comes from the same block. Higher values imply tighter within-block relations relative to between-block relations.

[b] ↑ means the higher the better, ↓ means the lower the better.

[c] Although the name is step-level, each node in the graph represents a sensor or an abnormality indicator instead of a step.

[d] In hours. Includes data simulation, block-level data processing and learning, step-level data processing and learning, aggregation, and evaluation.

Table 4.3: Simulation Results on the Effect of Hierarchical Modeling.

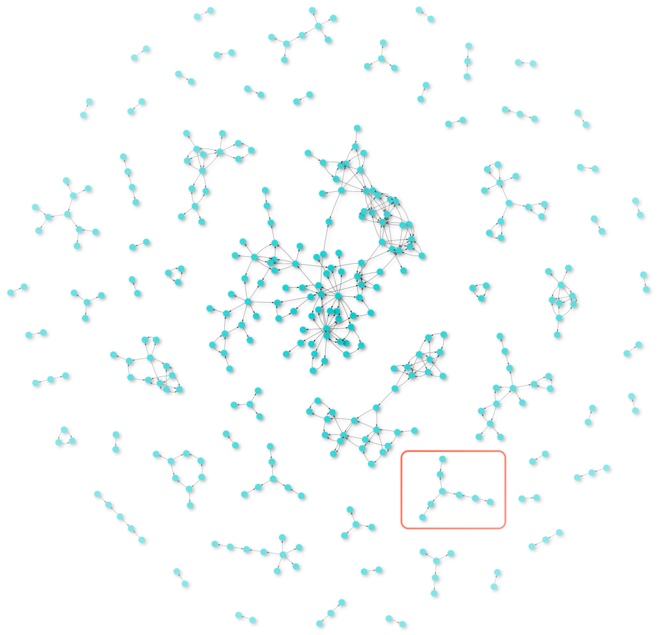


Figure 4.4: Ultimate Learned Causal Structure (strength ≥ 0.8).

could enhance the engineers' understanding of the manufacturing system with further investigation.

We also developed a zoomable network visualization tool that allows engineers and technicians to visualize and manipulate the causal structure. For example, Figure 4.4 displays the causal structure filtered with a strength threshold of 0.8, while Figure 4.5 shows a zoomed-in subgraph. Each node is labeled with a tool name (if it is a sensor node), stage, step, temporal order, and variable name, while each edge is labeled with a strength value indicating its credibility.

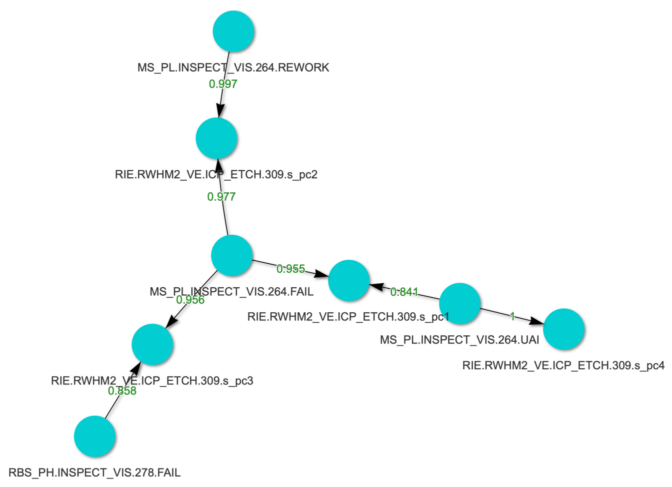

MS_PL.INSPECT_VIS.264.REWORK
0.997
RIE.RWHM2_VE.ICP_ETCH.309.s_pc2
0.977
0.955
MS_PL.INSPECT_VIS.264.FAIL
0.956
0.841
RIE.RWHM2_VE.ICP_ETCH.309.s_pc1
1
MS_PL.INSPECT_VIS.264.UAI
RIE.RWHM2_VE.ICP_ETCH.309.s_pc4
RIE.RWHM2_VE.ICP_ETCH.309.s_pc3
0.858
RBS_PH.INSPECT_VIS.278.FAIL

Figure 4.5: A Zoomed-in Subgraph of Figure 4.4. s_pc$k$ represents the kth principal component of sensors and the green numbers represent the strengths (credibility measure).

4.6 Conclusions and Discussions

This chapter presents a causal structure learning approach for wafer manufacturing data that addresses several challenges, such as mixed data types, missing entries, high dimensionality, data imbalance, and temporal and domain constraints. We propose three constrained mixed-data models and a hierarchical ensemble strategy to handle these challenges simultaneously. Our simulation studies demonstrate that the discretization-based method performs the best in accuracy and time cost and that the ensemble models are more effective at capturing weak signals with imbalanced data. Moreover, the hierarchical modeling strategy performs well under reasonable missing rates and produces robust fine-grained recall rates against the hierarchy setup. Finally, our hierarchical ensemble approach helps alleviate problems with data missing, high dimensionality, and data imbalance, and we have validated its effectiveness with an application to a wafer manufacturing data set.

Although we focus on wafer manufacturing, the challenges and data characteristics we address are not unique to this field. Therefore, our proposed approach can be applied to other domains with similar challenges. However, it is essential to note that our approach aims to balance all five challenges and may not be optimal for addressing a single aspect. Moreover, our proposal profoundly depends on the existence of temporal constraints. For example, the regression-based method and the union aggregation in the ensemble modeling would be invalid in situations without global temporal order. Additionally, when a high AR rate is a top priority, and the dimension is not very high, regression-based methods may be the preferred option at the cost of increasing computing time. Therefore, we recommend that practitioners carefully examine the data characteristics before selecting a modeling method.

We have observed that most existing work focuses on solving a single problem, whereas industrial applications often require integrated approaches that handle multiple aspects simultaneously. Therefore, further research is necessary to design integrated methods for such applications.

References

Ailem, M., Zhang, B., and Sha, F. (2019). Topic augmented generator for abstractive summarization. *arXiv preprint arXiv:1908.07026*.

Andrews, B., Ramsey, J., and Cooper, G. F. (2018). Scoring bayesian networks of mixed variables. *International Journal of Data Science and Analytics*, 6(1):3–18.

Andrews, B., Ramsey, J., and Cooper, G. F. (2019). Learning high-dimensional directed acyclic graphs with mixed data-types. In *The 2019 ACM SIGKDD Workshop on Causal Discovery*, pages 4–21. PMLR.

Azadkia, M. and Chatterjee, S. (2021). A simple measure of conditional dependence. *The Annals of Statistics*, 49(6):3070–3102.

Bahdanau, D., Cho, K. H., and Bengio, Y. (2015). Neural machine translation by jointly learning to align and translate. In *3rd International Conference on Learning Representations, ICLR 2015*.

Barnes, E. A., Samarasinghe, S. M., Ebert-Uphoff, I., and Furtado, J. C. (2019). Tropospheric and stratospheric causal pathways between the mjo and nao. *Journal of Geophysical Research: Atmospheres*, 124(16):9356–9371.

Belanche Muñoz, L. A. and Villegas, M. (2013). Kernel functions for categorical variables with application to problems in the life sciences. In *Artificial Intelligence Research and Development - Proceedings of the 16th International Conference of* *the Catalan Association for Artificial Intelligence, Vic, Catalonia, Spain, October 23-25, 2013*, volume 256 of *Frontiers in Artificial Intelligence and Applications*, pages 171–180. IOS Press.

Blei, D. M., Ng, A. Y., and Jordan, M. I. (2003). Latent dirichlet allocation. *Journal of machine Learning research*, 3(Jan):993–1022.

Bogachev, V. I. (2007). *Measure Theory*. Springer.

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*, pages 10–21, Berlin, Germany. Association for Computational Linguistics.

Breiman, L. (1996). Bagging predictors. *Machine learning*, 24(2):123–140.

Breiman, L. (2001). Random forests. *Machine learning*, 45:5–32.

Celikyilmaz, A., Bosselut, A., He, X., and Choi, Y. (2018). Deep Communicating Agents for Abstractive Summarization. In Walker, M. A., Ji, H., and Stent, A., editors, *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2018, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 1 (Long Papers)*, pages 1662–1675. Association for Computational Linguistics.

Chawla, N. V., Bowyer, K. W., Hall, L. O., and Kegelmeyer, W. P. (2002). Smote: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, 16:321–357.

Chen, X., Kingma, D. P., Salimans, T., Duan, Y., Dhariwal, P., Schulman, J., Sutskever, I., and Abbeel, P. (2017). Variational lossy autoencoder. In *5th In-*

ternational Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings. OpenReview.net.

Cheng, J. and Lapata, M. (2016). Neural Summarization by Extracting Sentences and Words. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 484–494.

Chickering, D. M. (2002). Optimal structure identification with greedy search. *Journal of machine learning research*, 3(Nov):507–554.

Chopra, S., Auli, M., and Rush, A. M. (2016). Abstractive sentence summarization with attentive recurrent neural networks. In *Proceedings of the 2016 conference of the North American chapter of the association for computational linguistics: human language technologies*, pages 93–98.

Cohan, A., Dernoncourt, F., Kim, D. S., Bui, T., Kim, S., Chang, W., and Goharian, N. (2018). A discourse-aware attention model for abstractive summarization of long documents. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 615–621, New Orleans, Louisiana. Association for Computational Linguistics.

Colombo, D., Maathuis, M. H., et al. (2014). Order-independent constraint-based causal structure learning. *J. Mach. Learn. Res.*, 15(1):3741–3782.

Cui, R., Groot, P., and Heskes, T. (2016). Copula pc algorithm for causal discovery from mixed data. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 377–392. Springer.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2019). BERT: Pre-training of deep bidirectional transformers for language understanding. 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)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Dieng, A. B., Ruiz, F. J., and Blei, D. (2020). Topic Modeling in Embedding Spaces. *Transactions of the Association for Computational Linguistics*, 8:439–453.

Dieng, A. B., Wang, C., Gao, J., and Paisley, J. W. (2017). TopicRNN: A Recurrent Neural Network with Long-Range Semantic Dependency. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Dinh, L., Krueger, D., and Bengio, Y. (2015). NICE: non-linear independent components estimation. In Bengio, Y. and LeCun, Y., editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Workshop Track Proceedings*.

Dinh, L., Sohl-Dickstein, J., and Bengio, S. (2017). Density estimation using real NVP. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Dolatabadi, H. M., Erfani, S. M., and Leckie, C. (2020). Invertible generative modeling using linear rational splines. In Chiappa, S. and Calandra, R., editors, *The 23rd International Conference on Artificial Intelligence and Statistics, AISTATS 2020, 26-28 August 2020, Online [Palermo, Sicily, Italy]*, volume 108 of *Proceedings of Machine Learning Research*, pages 4236–4246. PMLR.

Dong, L., Yang, N., Wang, W., Wei, F., Liu, X., Wang, Y., Gao, J., Zhou, M., and Hon, H. (2019). Unified language model pre-training for natural language understanding and generation. In Wallach, H. M., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E. B., and Garnett, R., editors, *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 13042–13054.

Du, W., Zhao, J., Wang, L., and Ji, Y. (2022). Diverse text generation via variational encoder-decoder models with gaussian process priors. *arXiv preprint arXiv:2204.01227*.

Durkan, C., Bekasov, A., Murray, I., and Papamakarios, G. (2019). Neural spline flows. In Wallach, H. M., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E. B., and Garnett, R., editors, *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 7509–7520.

Fabbri, A., Li, I., She, T., Li, S., and Radev, D. (2019a). Multi-news: A large-scale multi-document summarization dataset and abstractive hierarchical model. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1074–1084, Florence, Italy. Association for Computational Linguistics.

Fabbri, A. R., Li, I., She, T., Li, S., and Radev, D. (2019b). Multi-News: A Large-Scale Multi-Document Summarization Dataset and Abstractive Hierarchical Model. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1074–1084.

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)*, pages 889–898.

Fan, J. and Li, R. (2001). Variable selection via nonconcave penalized likelihood and its oracle properties. *Journal of the American statistical Association*, 96(456):1348–1360.

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 Burstein, J., Doran, C., and Solorio, T., editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 240–250. Association for Computational Linguistics.

Fu, X., Wang, J., Zhang, J., Wei, J., and Yang, Z. (2020). Document summarization with VHTM: variational hierarchical topic-aware mechanism. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 7740–7747. AAAI Press.

Fu, Z., Lam, W., So, A. M., and Shi, B. (2021). A theoretical analysis of the repetition problem in text generation. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*, pages 12848–12856. AAAI Press.

Gao, E., Ng, I., Gong, M., Shen, L., Huang, W., Liu, T., Zhang, K., and Bondell, H. D. (2022). Missdag: Causal discovery in the presence of missing data with continuous additive noise models. *CoRR*, abs/2205.13869.

Gehrmann, S., Deng, Y., and Rush, A. M. (2018). Bottom-Up Abstractive Summarization. In Riloff, E., Chiang, D., Hockenmaier, J., and Tsujii, J., editors, *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 4098–4109. Association for Computational Linguistics.

Germain, M., Gregor, K., Murray, I., and Larochelle, H. (2015). MADE: masked autoencoder for distribution estimation. In Bach, F. R. and Blei, D. M., editors, *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*, volume 37 of *JMLR Workshop and Conference Proceedings*, pages 881–889. JMLR.org.

Gharahbagheri, H., Imtiaz, S., Khan, F., and Ahmed, S. (2015). Causality analysis for root cause diagnosis in fluid catalytic cracking unit. *IFAC-PapersOnLine*, 48(21):838–843.

Gliwa, B., Mochol, I., Biesek, M., and Wawer, A. (2019). SAMSum corpus: A human-annotated dialogue dataset for abstractive summarization. In *Proceedings of the 2nd Workshop on New Frontiers in Summarization*, pages 70–79, Hong Kong, China. Association for Computational Linguistics.

Glorot, X. and Bengio, Y. (2010). Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 249–256. JMLR Workshop and Conference Proceedings.

Glover, F. (1989). Tabu search—part i. *ORSA Journal on Computing*, 1(3):190–206.

Glover, F. (1990). Tabu search—part ii. *ORSA Journal on Computing*, 2(1):4–32.

Glymour, C., Zhang, K., and Spirtes, P. (2019). Review of causal discovery methods based on graphical models. *Frontiers in genetics*, 10:524.

Grusky, M., Naaman, M., and Artzi, Y. (2018). Newsroom: A Dataset of 1.3 Million Summaries with Diverse Extractive Strategies. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 708–719.

Gu, A., Gülçehre, Ç., Paine, T., Hoffman, M., and Pascanu, R. (2020). Improving the gating mechanism of recurrent neural networks. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pages 3800–3809. PMLR.

Handhayani, T. and Cussens, J. (2020). Kernel-based approach for learning causal graphs from mixed data. In *International Conference on Probabilistic Graphical Models*, pages 221–232. PMLR.

He, J., Spokoyny, D., Neubig, G., and Berg-Kirkpatrick, T. (2019). Lagging inference networks and posterior collapse in variational autoencoders. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Heckerman, D., Geiger, D., and Chickering, D. M. (1995). Learning bayesian networks: The combination of knowledge and statistical data. *Machine learning*, 20(3):197–243.

Hermann, K. M., Kociský, T., Grefenstette, E., Espeholt, L., Kay, W., Suleyman, M., and Blunsom, P. (2015). Teaching machines to read and comprehend. In Cortes, C., Lawrence, N. D., Lee, D. D., Sugiyama, M., and Garnett, R., editors, *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pages 1693–1701.

Hinton, G., Vinyals, O., and Dean, J. (2015). Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*.

Holtzman, A., Buys, J., Du, L., Forbes, M., and Choi, Y. (2020). The curious case of neural text degeneration. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Huegle, J., Hagedorn, C., and Uflacker, M. (2020). How causal structural knowledge adds decision-support in monitoring of automotive body shop assembly lines. In Bessiere, C., editor, *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20*. International Joint Conferences on Artificial Intelligence Organization.

Hyvarinen, A. (1999). Fast and robust fixed-point algorithms for independent component analysis. *IEEE transactions on Neural Networks*, 10(3):626–634.

Kim, Y., Wiseman, S., Miller, A. C., Sontag, D. A., and Rush, A. M. (2018). Semi-amortized variational autoencoders. In Dy, J. G. and Krause, A., editors, *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80 of *Proceedings of Machine Learning Research*, pages 2683–2692. PMLR.

Kingma, D. P. and Ba, J. (2015). Adam: A method for stochastic optimization. In Bengio, Y. and LeCun, Y., editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Kingma, D. P., Salimans, T., Jozefowicz, R., Chen, X., Sutskever, I., and Welling, M. (2016). Improved variational inference with inverse autoregressive flow. In Lee, D., Sugiyama, M., Luxburg, U., Guyon, I., and Garnett, R., editors, *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc.

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

Krawczyk, B. (2016). Learning from imbalanced data: open challenges and future directions. *Progress in Artificial Intelligence*, 5(4):221–232.

Kyono, T., Zhang, Y., Bellot, A., and van der Schaar, M. (2021). Miracle: Causally-aware imputation via learning missing data mechanisms. *Advances in Neural Information Processing Systems*, 34:23806–23817.

Landman, R. and Jämsä-Jounela, S.-L. (2016). Hybrid approach to casual analysis on a complex industrial system based on transfer entropy in conjunction with process connectivity information. *Control Engineering Practice*, 53:14–23.

Lee, J. D. and Hastie, T. J. (2015). Learning the structure of mixed graphical models. *Journal of Computational and Graphical Statistics*, 24(1):230–253.

Lewis, M., Liu, Y., Goyal, N., Ghazvininejad, M., Mohamed, A., Levy, O., Stoyanov, V., and Zettlemoyer, L. (2020). BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Li, C., Shen, X., and Pan, W. (2023). Inference for a large directed acyclic graph with unspecified interventions. *Journal of Machine Learning Research*, 24(73):1–48.

Liang, S. Y., Hecker, R. L., and Landers, R. G. (2004). Machining process monitoring and control: the state-of-the-art. *J. Manuf. Sci. Eng.*, 126(2):297–310.

Lin, C.-Y. (2004a). Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81.

Lin, C.-Y. (2004b). ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Little, R. J. and Rubin, D. B. (2019). *Statistical analysis with missing data*, volume 793. John Wiley & Sons.

Liu, Y., Chen, X., Luo, X., and Zhu, K. Q. (2021). Reducing repetition in convolutional abstractive summarization. *Natural Language Engineering*, pages 1–29.

Liu, Y. and Lapata, M. (2019a). Text Summarization with Pretrained Encoders. 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)*, pages 3721–3731.

Liu, Y. and Lapata, M. (2019b). Text summarization with pretrained encoders. 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)*, pages 3730–3740, Hong Kong, China. Association for Computational Linguistics.

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., and Stoyanov, V. (2019). Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Luo, T. and Chien, J. (2021). Variational dialogue generation with normalizing flows. In *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2021, Toronto, ON, Canada, June 6-11, 2021*, pages 7778–7782. IEEE.

Maxwell Chickering, D. and Heckerman, D. (1997). Efficient approximations for the marginal likelihood of bayesian networks with hidden variables. *Machine learning*, 29(2):181–212.

Meinshausen, N. and Bühlmann, P. (2006). High-dimensional graphs and variable selection with the lasso. *Annals of statistics*, 34(3):1436–1462.

Mi, H., Sankaran, B., Wang, Z., and Ittycheriah, A. (2016). Coverage Embedding Models for Neural Machine Translation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 955–960.

Nair, P. and Singh, A. K. (2021). On Reducing Repetition in Abstractive Summarization. In *Proceedings of the Student Research Workshop Associated with RANLP 2021*, pages 126–134.

Nallapati, R., Zhai, F., and Zhou, B. (2017). Summarunner: A recurrent neural network based sequence model for extractive summarization of documents. In Singh, S. and Markovitch, S., editors, *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA*, pages 3075–3081. AAAI Press.

Nallapati, R., Zhou, B., dos Santos, C. N., Gülçehre, Ç., and Xiang, B. (2016). Abstractive text summarization using sequence-to-sequence rnns and beyond. In Goldberg, Y. and Riezler, S., editors, *Proceedings of the 20th SIGNLL Conference on Computational Natural Language Learning, CoNLL 2016, Berlin, Germany, August 11-12, 2016*, pages 280–290. ACL.

Nandy, P., Hauser, A., and Maathuis, M. H. (2018). High-dimensional consistency in score-based and hybrid structure learning. *The Annals of Statistics*, 46(6A):3151–3183.

Narayan, S., Cohen, S. B., and Lapata, M. (2018a). Don't give me the details, just the summary! topic-aware convolutional neural networks for extreme summarization. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1797–1807, Brussels, Belgium. Association for Computational Linguistics.

Narayan, S., Cohen, S. B., and Lapata, M. (2018b). Don't Give Me the Details, Just the Summary! Topic-Aware Convolutional Neural Networks for Extreme Summarization. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1797–1807.

Nguyen, T., Luu, A. T., Lu, T., and Quan, T. (2021). Enriching and controlling global semantics for text summarization. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 9443–9456, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Papamakarios, G., Murray, I., and Pavlakou, T. (2017). Masked autoregressive flow for density estimation. In Guyon, I., von Luxburg, U., Bengio, S., Wallach, H. M., Fergus, R., Vishwanathan, S. V. N., and Garnett, R., editors, *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 2338–2347.

Paulus, R., Xiong, C., and Socher, R. (2018). A Deep Reinforced Model for Abstractive Summarization. In *6th International Conference on Learning Representations,* *ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net.

Pearl, J. (2009). *Causality*. Cambridge University Press, 2 edition.

Prokhorov, V., Shareghi, E., Li, Y., Pilehvar, M. T., and Collier, N. (2019). On the importance of the Kullback-Leibler divergence term in variational autoencoders for text generation. In *Proceedings of the 3rd Workshop on Neural Generation and Translation*, pages 118–127, Hong Kong. Association for Computational Linguistics.

Qi, W., Yan, Y., Gong, Y., Liu, D., Duan, N., Chen, J., Zhang, R., and Zhou, M. (2020). ProphetNet: Predicting future n-gram for sequence-to-SequencePre-training. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2401–2410, Online. Association for Computational Linguistics.

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., and Sutskever, I. (2019). Language models are unsupervised multitask learners. *OpenAI Blog*, 1(8):9.

Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., and Liu, P. J. (2020). Exploring the limits of transfer learning with a unified text-to-text transformer. *The Journal of Machine Learning Research*, 21(1):5485–5551.

Ramsey, J., Glymour, M., Sanchez-Romero, R., and Glymour, C. (2017). A million variables and more: the fast greedy equivalence search algorithm for learning high-dimensional graphical causal models, with an application to functional magnetic resonance images. *International journal of data science and analytics*, 3(2):121–129.

Ranzato, M., Chopra, S., Auli, M., and Zaremba, W. (2016). Sequence level training with recurrent neural networks. In Bengio, Y. and LeCun, Y., editors, *4th Inter-*

national Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings.

Reimers, N. and Gurevych, I. (2019). Sentence-BERT: Sentence embeddings using Siamese BERT-networks. 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)*, pages 3982–3992, Hong Kong, China. Association for Computational Linguistics.

Rezende, D. J. and Mohamed, S. (2015). Variational inference with normalizing flows. In Bach, F. R. and Blei, D. M., editors, *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*, volume 37 of *JMLR Workshop and Conference Proceedings*, pages 1530–1538. JMLR.org.

Rombach, R., Blattmann, A., Lorenz, D., Esser, P., and Ommer, B. (2022). High-resolution image synthesis with latent diffusion models. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022, New Orleans, LA, USA, June 18-24, 2022*, pages 10674–10685. IEEE.

Rothe, S., Narayan, S., and Severyn, A. (2020a). Leveraging pre-trained checkpoints for sequence generation tasks. *Transactions of the Association for Computational Linguistics*, 8:264–280.

Rothe, S., Narayan, S., and Severyn, A. (2020b). Leveraging pre-trained checkpoints for sequence generation tasks. *Transactions of the Association for Computational Linguistics*, 8:264–280.

Runge, J., Bathiany, S., Bollt, E., Camps-Valls, G., Coumou, D., Deyle, E., Glymour, C., Kretschmer, M., Mahecha, M. D., Muñoz-Marí, J., et al. (2019). Inferring causation from time series in earth system sciences. *Nature communications*, 10(1):2553.

Rush, A. M., Chopra, S., and Weston, J. (2015). A Neural Attention Model for Abstractive Sentence Summarization. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 379–389.

Sankaran, B., Mi, H., Al-Onaizan, Y., and Ittycheriah, A. (2016). Temporal attention model for neural machine translation. *arXiv preprint arXiv:1608.02927*.

Scutari, M. and Denis, J.-B. (2021). *Bayesian networks: with examples in R*. Chapman and Hall/CRC.

Sedgewick, A. J., Buschur, K., Shi, I., Ramsey, J. D., Raghu, V. K., Manatakis, D. V., Zhang, Y., Bon, J., Chandra, D., Karoleski, C., et al. (2019). Mixed graphical models for integrative causal analysis with application to chronic lung disease diagnosis and prognosis. *Bioinformatics*, 35(7):1204–1212.

See, A., Liu, P. J., and Manning, C. D. (2017a). Get to the point: Summarization with pointer-generator networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1073–1083, Vancouver, Canada. Association for Computational Linguistics.

See, A., Liu, P. J., and Manning, C. D. (2017b). Get To The Point: Summarization with Pointer-Generator Networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1073–1083.

Semeniuta, S., Severyn, A., and Barth, E. (2017). A hybrid convolutional variational autoencoder for text generation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 627–637, Copenhagen, Denmark. Association for Computational Linguistics.

Serban, I. V., Sordoni, A., Lowe, R., Charlin, L., Pineau, J., Courville, A. C., and Bengio, Y. (2017). A hierarchical latent variable encoder-decoder model for generating dialogues. In Singh, S. and Markovitch, S., editors, *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA*, pages 3295–3301. AAAI Press.

Setiawan, H., Sperber, M., Nallasamy, U., and Paulik, M. (2020). Variational neural machine translation with normalizing flows. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7771–7777, Online. Association for Computational Linguistics.

Shah, S. Y., Dang, X.-H., and Zerfos, P. (2018). Root cause detection using dynamic dependency graphs from time series data. In *2018 IEEE International Conference on Big Data (Big Data)*, pages 1998–2003. IEEE.

Shen, D., Zhang, Y., Henao, R., Su, Q., and Carin, L. (2018). Deconvolutional latent-variable model for text sequence matching. In McIlraith, S. A. and Weinberger, K. Q., editors, *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, pages 5438–5445. AAAI Press.

Shen, X., Pan, W., and Zhu, Y. (2012). Likelihood-based selection and sharp parameter estimation. *Journal of the American Statistical Association*, 107(497):223–232.

Shen, X., Pan, W., Zhu, Y., and Zhou, H. (2013). On constrained and regularized high-dimensional regression. *Annals of the Institute of Statistical Mathematics*, 65(5):807–832.

Shimizu, S., Hoyer, P. O., Hyvärinen, A., Kerminen, A., and Jordan, M. (2006). A linear non-gaussian acyclic model for causal discovery. *Journal of Machine Learning Research*, 7(10).

Shleifer, S. and Rush, A. M. (2020). Pre-trained summarization distillation. *arXiv preprint arXiv:2010.13002*.

Sokolova, E., von Rhein, D., Naaijen, J., Groot, P., Claassen, T., Buitelaar, J., and Heskes, T. (2017). Handling hybrid and missing data in constraint-based causal discovery to study the etiology of adhd. *International journal of data science and analytics*, 3:105–119.

Song, K., Tan, X., Qin, T., Lu, J., and Liu, T. (2019). MASS: masked sequence to sequence pre-training for language generation. In Chaudhuri, K. and Salakhutdinov, R., editors, *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pages 5926–5936. PMLR.

Spirtes, P., Glymour, C. N., Scheines, R., and Heckerman, D. (2000). *Causation, prediction, and search*. MIT Press.

Städler, N. and Bühlmann, P. (2012). Missing values: sparse inverse covariance estimation and an extension to sparse regression. *Statistics and Computing*, 22:219–235.

Su, J., Wu, S., Xiong, D., Lu, Y., Han, X., and Zhang, B. (2018). Variational recurrent neural machine translation. In McIlraith, S. A. and Weinberger, K. Q., editors, *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence*

(EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018, pages 5488–5495. AAAI Press.

Tabak, E. G. and Turner, C. V. (2013). A family of nonparametric density estimation algorithms. *Communications on Pure and Applied Mathematics*, 66(2):145–164.

Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B (Methodological)*, 58(1):267–288.

Tolstikhin, I. O., Bousquet, O., Gelly, S., and Schölkopf, B. (2018). Wasserstein auto-encoders. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net.

Tran, D., Vafa, K., Agrawal, K. K., Dinh, L., and Poole, B. (2019). Discrete flows: Invertible generative models of discrete data. In Wallach, H. M., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E. B., and Garnett, R., editors, *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 14692–14701.

Tsagris, M., Borboudakis, G., Lagani, V., and Tsamardinos, I. (2018). Constraint-based causal discovery with mixed data. *International journal of data science and analytics*, 6(1):19–30.

Tsamardinos, I., Brown, L. E., and Aliferis, C. F. (2006). The max-min hill-climbing bayesian network structure learning algorithm. *Machine learning*, 65(1):31–78.

Tu, R., Zhang, C., Ackermann, P., Mohan, K., Kjellström, H., and Zhang, K. (2019). Causal discovery in the presence of missing data. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 1762–1770. PMLR.

Tu, Z., Lu, Z., Liu, Y., Liu, X., and Li, H. (2016). Modeling Coverage for Neural Machine Translation. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 76–85.

van den Berg, R., Hasenclever, L., Tomczak, J. M., and Welling, M. (2018). Sylvester normalizing flows for variational inference. In Globerson, A. and Silva, R., editors, *Proceedings of the Thirty-Fourth Conference on Uncertainty in Artificial Intelligence, UAI 2018, Monterey, California, USA, August 6-10, 2018*, pages 393–402. AUAI Press.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. (2017). Attention is all you need. In Guyon, I., von Luxburg, U., Bengio, S., Wallach, H. M., Fergus, R., Vishwanathan, S. V. N., and Garnett, R., editors, *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008.

Wang, L., Yao, J., Tao, Y., Zhong, L., Liu, W., and Du, Q. (2018). A reinforced topic-aware convolutional sequence-to-sequence model for abstractive text summarization. *arXiv preprint arXiv:1805.03616*.

Wang, W., Gan, Z., Xu, H., Zhang, R., Wang, G., Shen, D., Chen, C., and Carin, L. (2019). Topic-guided variational auto-encoder for text 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)*, pages 166–177, Minneapolis, Minnesota. Association for Computational Linguistics.

Wang, Z., Duan, Z., Zhang, H., Wang, C., Tian, L., Chen, B., and Zhou, M. (2020a). Friendly topic assistant for transformer based abstractive summarization. In *Pro-*

ceedings of the 2020 conference on empirical methods in natural language processing (EMNLP), pages 485–497.

Wang, Z., Duan, Z., Zhang, H., Wang, C., Tian, L., Chen, B., and Zhou, M. (2020b). Friendly topic assistant for transformer based abstractive summarization. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 485–497, Online. Association for Computational Linguistics.

Welleck, S., Kulikov, I., Roller, S., Dinan, E., Cho, K., and Weston, J. (2019). Neural Text Generation With Unlikelihood Training. In *International Conference on Learning Representations*.

Yang, Y. (2023). Simulation Results on the Effect of Ensemble on Data Imbalance.

Yang, Z., Hu, Z., Salakhutdinov, R., and Berg-Kirkpatrick, T. (2017). Improved variational autoencoders for text modeling using dilated convolutions. In Precup, D. and Teh, Y. W., editors, *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, volume 70 of *Proceedings of Machine Learning Research*, pages 3881–3890. PMLR.

Zhang, B., Xiong, D., Su, J., Duan, H., and Zhang, M. (2016). Variational neural machine translation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 521–530, Austin, Texas. Association for Computational Linguistics.

Zhang, C.-H. (2010). Nearly unbiased variable selection under minimax concave penalty. *The Annals of statistics*, 38(2):894–942.

Zhang, J., Zhao, Y., Saleh, M., and Liu, P. J. (2020a). PEGASUS: pre-training with extracted gap-sentences for abstractive summarization. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020,* *Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pages 11328–11339. PMLR.

Zhang, T., Kishore, V., Wu, F., Weinberger, K. Q., and Artzi, Y. (2020b). Bertscore: Evaluating text generation with BERT. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Zhao, S., Song, J., and Ermon, S. (2017). Infovae: Information maximizing variational autoencoders. *arXiv preprint arXiv:1706.02262*.

Zheng, C., Zhang, K., Wang, H. J., Fan, L., and Wang, Z. (2020). Topic-guided abstractive text summarization: a joint learning approach. *arXiv preprint arXiv:2010.10323*.

Zhou, C. and Neubig, G. (2017). Morphological inflection generation with multi-space variational encoder-decoders. In *Proceedings of the CoNLL SIGMORPHON 2017 Shared Task: Universal Morphological Reinflection*, pages 58–65, Vancouver. Association for Computational Linguistics.

Zhu, Y., Shen, X., and Pan, W. (2020). On high-dimensional constrained maximum likelihood inference. *Journal of the American Statistical Association*, 115(529):217–230.

Zou, Y., Zhao, L., Kang, Y., Lin, J., Peng, M., Jiang, Z., Sun, C., Zhang, Q., Huang, X., and Liu, X. (2021). Topic-oriented spoken dialogue summarization for customer service with saliency-aware topic modeling. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*, pages 14665–14673. AAAI Press.

Appendix A

Softwares

A.1 R Package: glmtlp

We build on the work of Li et al. (2023) to develop a publicly available R package **glmtlp**, providing easy and efficient implementations of constrained L_0 penalty and its surrogate TLP (Shen et al., 2012) under various settings, including linear regression, logistic regression, undirected graph, and directed acyclic graph. In particular, we provide functions to conduct statistical inference (Zhu et al., 2020), focusing on either a specific variable or an arbitrary set of variables. In this package, the constrained L_0-likelihood is optimized by a new algorithm called the projection DC programming (Li et al., 2023), which enjoys several good properties. First, the projection DC programming establishes a connection between the constrained problem and the regularized problem. As a result, it allows an efficient implementation with the warm start trick, greatly boosting the computation speed. Second, the projection DC programming is shown to deliver the global minimum with a probability tending to one with respect to the data generating distribution (Li et al., 2023). Third, the package integrates the estimation and inference procedures, permitting both individual-level data matrices and summary statistics as the input, providing a user-friendly and flexible interface for various applications. Furthermore, we implement several tricks including sequential screening rule, warm start, and parallelism, and coordinate marjorization descent to make the algorithm fast and efficient. The package is available from the Comprehensive R Archive Network (CRAN) at `https://cran.r-project.org/web/packages/glmtlp/index.html`.

A.2 Python Package: flowsum

I have created a Python package, **flowsum**, that encapsulates the FlowSUM framework detailed in Chapter 3. This package is organized into four sub-modules. The `models` sub-module establishes the NF latent module and the NF-enhanced Transformer-based models. The `nf` sub-module articulates the transformations of normalizing flows. `trainers` provides trainers that facilitate BOWs input, output gatescore, perplexity, KL divergence, and enable CAAT, also featuring callbacks for latent distribution visualization. Lastly, the `utils` sub-module offers utility functions related to data processing and loading, model evaluation, training logging, model and argument specification, and visualization. The package can be accessed at Github[1].

A.3 R & Python Package: CausalLearn & causal-learn

During my collaboration with Seagate Technology, I created an R package called **CausalLearn**. This package implements mixed data causal structure learning methods, including discretization-based learning, regression-based learning, and CKAPC, as described in Section 4.4.3. Additionally, the package includes a visualization tool that enables engineers and technicians to easily manipulate the zoomable causal struc-

[1]Currently, it is exclusively available to UMN. It will be made public upon receiving the decision regarding the FlowSUM paper from EMNLP 2023.

tures it generates.

Furthermore, I developed a Python package called **causallearn** to implement the entire modeling pipeline shown in Figure 4.3. This pipeline involves querying data from databases, pre-processing sensor and measurement data, using functions from the R package **CausalLearn** to learn causal structures, and automatically generating a report with the learned structure. However, due to business confidentiality, these two packages are not publicly available.

Appendix B

Chapter 2 Appendices

B.1 Training and Testing Details

For all models, batch size is 16, beam size is 5, dropout rate is 0.2, hidden size is 300, L_2 weight decay rate is 1.2×10^{-6} in the Adam optimizer. The other hyper-parameters are as shown in Table B.1. The parameters in the linear layers are initialized as $N(0, 10^{-4})$. And for the LSTM layers, the weight parameters are initialized with the Xavier initialization method (Glorot and Bengio, 2010) and the bias parameters are initialized as zeros. The learning rate decays exponentially along with the steps before reaching a minimum threshold, as defined below.

$$LR = \max\left(0.999^{\text{steps}/10}, 0.002\right)$$

Note that in the CNN/Daily Mail dataset, the initial learning rate and the max gradient norm are different across the models. The main reason is that as the model grows more complex, large initial learning rate and max gradient norm would lead to NaNs during training. The ultimate choices of these two hyper-parameters are established by experimenting till there occur no NaN values during training.

For the sake of time efficiency, we use a subset of the validation set to evaluate the model performance during training and the perplexity on the validation subset is used to select the best checkpoint and to perform early stopping. And it is the early stopping mechanism that causes the number of training epochs differ across models.

CNN/Daily Mail								
Model	Initial Learning rate	Num of epochs	Max gradient norm	Num of topics	Max input tokens	Max Target tokens	Validation subset size	Max decoding steps
Baseline	0.015	20	2.0	-	512	-[a]	1024	100
Baseline + enc_penalty	0.01	20	2.0	-	512	-	1024	100
BTM	0.015	25	1.0	200	512	-	1024	100
BTM + enc_penalty	0.01	35	1.0	200	512	-	1024	100
TAM	0.015	25	1.0	200	512	-	1024	100
TAM + topic_penalty	0.008	35	1.0	200	512	-	1024	100
TAM + enc_penalty	0.01	35	1.0	200	512	-	1024	100
PA-TAM	0.008	35	1.0	200	512	-	1024	100
Multi-News								
Model	Initial Learning rate	Num of epochs	Max gradient norm	Num of topics	Max input tokens	Max Target tokens	Validation subset size	Max decoding steps
Baseline	0.008	150	1.0	-	500	256	1024	256
TAM	0.008	90	1.0	150	500	256	1024	256
PA-Baseline	0.006	100	1.0	-	500	256	1024	256
PA-TAM	0.008	50	1.0	150	500	256	1024	256
NEWSROOM								
Model	Initial Learning rate	Num of epochs	Max gradient norm	Num of topics	Max input tokens	Max Target tokens	Validation subset size	Max decoding steps
Baseline	0.008	15	1.0	-	512	-	8192	100
TAM	0.008	15	1.0	150	512	-	8192	100
PA-Baseline	0.008	15	1.0	-	512	-	8192	100
PA-TAM	0.008	15	1.0	150	512	-	8192	100

[a] "-" means not applicable. For example, the baseline model does not have a topic module, so the Num of topics entry is "-".

Table B.1: Hyper-parameters of the training and testing stages reported in Section 2.4.

Appendix C

Chapter 3 Appendices

C.1 Derivation of ELBO

$$
\begin{aligned}
&\mathrm{KL}(q(z \mid x, y) \| p(z \mid x, y)) \\
=&\mathbb{E}_{q(z|x,y)}[\log q(z \mid x, y)] - \mathbb{E}_{q(z|x,y)}\left[\log \frac{p(z, x, y)}{p(x, y)}\right] \\
=&\mathbb{E}_{q(z|x,y)}[\log q(z \mid x, y)] - \mathbb{E}_{q(z|x,y)}\left[\log \frac{p(z, x, y)}{p(x, z)} \cdot \frac{p(x, z)}{p(x)} \cdot \frac{p(x)}{p(x, y)}\right] \\
=&\mathbb{E}_{q(z|x,y)}[\log q(z \mid x, y)] - \mathbb{E}_{q(z|x,y)}[\log p(y \mid x, z)] - \mathbb{E}_{q(z|x,y)}[\log p(z \mid x)] \\
&+ \mathbb{E}_{q(z|x,y)}[\log p(y \mid x)] \\
=&KL(q(z \mid x, y) \| p(z \mid x)) - \mathbb{E}_{q(z|x,y)}[\log p(y \mid x, z)] + \mathbb{E}_{q(z|x,y)}[\log p(y \mid x)] \\
\geqslant&0 \\
\\
\Rightarrow&\mathrm{ELBO}_{\mathrm{VED}} \\
=&\mathbb{E}_{q(z|x,y)}[\log p(y \mid x, z)] - KL(q(z \mid x, y) \| p(z \mid x)) \\
\leq&\log p(y \mid x)
\end{aligned}
\tag{C.1}
$$

$$\begin{aligned}
&\mathrm{ELBO}_{\text{NF-VED}} \\
=&\mathbb{E}_{q(z|x)}[\log p(y \mid x, z)] + \mathbb{E}_{q(z|x)} \log p(z \mid x) - \mathbb{E}_{q(z|x)}[\log q(z \mid x)] \\
=&\mathbb{E}_{q_0(z_0)}\left[\log p\left(y \mid x, z_K\right) + \log p\left(z_K \mid x\right)\right] - \mathbb{E}_{q_0(z_0)}\left[\log q_K\left(z_K\right)\right] \\
=&\mathbb{E}_{q_0(z_0)}\Big[\log p\left(y \mid x, z_K\right) + \log p\left(z_K \mid x\right)\Big] \\
&- \mathbb{E}_{q_0(z_0)}\left[\log q_0\left(z_0\right) - \sum_{k=1}^{K} \log\left|\det J_{f_k}\left(z_{k-1}\right)\right|\right],
\end{aligned} \tag{C.2}$$

where q_0 and q_K are the probability density function for z_0 and z_K respectively.

C.2 Implementation Details

C.2.1 NF Latent Module

We configure the inference net $q(z_0|\overline{x})$ to be a feedforward neural network with three hidden layers of dimension $\in \{300, 600\}$, Tanh activations, and a 0.1 dropout rate. We set the latent dimension ℓ to 300 and the number of NF layers $\in \{2, 4, 6, 8\}$. For spline coupling layers (both RLNSF and RQNSF), we set the number of bins to 4, the bound to 3.0, the split dimension to $\ell/2$, and the neural network to have two hidden layers with the dimension ℓ. For RealNVP, the split dimension is $\ell/2$, and the neural network has one hidden layer with a dimension of 10ℓ. For IAF, the neural network features one hidden layer of the dimension $3\ell+1$. Moreover, we set $\beta = 1$ and $C = 0.1$ for models that use β_C-VAE, and for models that use CAAT, we conduct one epoch of aggressive training with $n_{alt} = 15$, followed by two epochs of non-aggressive training.

C.2.2 Optimization

We train the models using the Adam optimizer (Kingma and Ba, 2015) with $\beta_1 = 0.9, \beta_2 = 0.999$, and $\epsilon = 10^{-8}$. The initial learning rate is 5×10^{-5}. We employ a linear learning rate scheduler that increases the learning rate from 0 to the initial learning rate during the warmup stage and decreases it from the initial learning rate to 0 after the warmup stage. We also apply the gradient clipping technique with a maximum gradient norm of 1.0. Furthermore, we terminate the training early when the perplexity fails to improve for eight or sixteen consecutive evaluation calls.

C.2.3 Model Hyper Parameters

Table C.1 provides the hyper-parameters for the models discussed in Table 3.4 - Table 3.7, for the sake of reproducibility. To ensure fair comparisons, unless otherwise specified, the VAESUM models typically employ the same set of hyper-parameters as their FlowSUM counterparts, except with standard training and no NF layers applied. Additionally, the models in Table 3.8 have the same hyper-parameters as those in Table 3.7, except for the number of NF layers used. Lastly, in Table 3.9, all FlowSUM models use 4 NF layers and the same set of hyper-parameters as those in Table 3.7, but vary in their training strategies.

FlowSUM in Table 3.4

Dataset	Number of epochs	Number of aggressive epochs	Batch size	Inference net hidden dim	NF type	Number of NF layers	Beam size	Length penalty	Max input tokens	Max target tokens
CNN/Daily Mail	3	1	8	300	RQNSF	4	4	2.0	1024	128
Multi-News	3	1	8	600	IAF	6	4	2.0	1024	128
arXiv	4	1	16	600	RQNSF	4	4	2.0	1024	142
PubMed	4	1	16	600	RQNSF	6	4	2.0	1024	142
XSum	3	1	8	600	RQNSF	4	6	0.5	1024	62
SAMSum	12	12	8	600	RQNSF	4	6	1.0	1024	62

Models in Table 3.5

Model	Number of epochs	Number of aggressive epochs	Batch size	Inference net hidden dim	NF type	Number of NF layers	Beam size	Length penalty	Max input tokens	Max target tokens
VAESUM	3	0	8	600	-[a]	-	4	2.0	1024	128
FlowSUM (Planar)	3	0	8	600	Planar	4	4	2.0	1024	128
FlowSUM (RQNSF)	3	1	8	300	RQNSF	4	4	2.0	1024	128
BART-PLKD	3	0	8	-	-	-	4	2.0	1024	128
VAESUM-PLKD	3	0	8	600	-	-	4	2.0	1024	128
FlowSUM-PLKD (Planar)	3	0	8	600	Planar	4	4	2.0	1024	128
FlowSUM-PLKD (RQNSF)	3	1	8	300	RQNSF	4	4	2.0	1024	128

Models in Table 3.6

Model	Number of epochs	Number of aggressive epochs	Batch size	Inference net hidden dim	NF type	Number of NF layers	Beam size	Length penalty	Max input tokens	Max target tokens
dBART-6-6										
FlowSUM	3	1	8	300	RQNSF	4	4	2.0	1024	128
FlowSUM-PLKD	3	1	8	300	RQNSF	4	4	2.0	1024	128
dBART-12-3										
FlowSUM	3	1	8	300	RQNSF	4	4	2.0	1024	128
FlowSUM-PLKD	3	1	8	300	RQNSF	4	4	2.0	1024	128

Models in Table 3.7

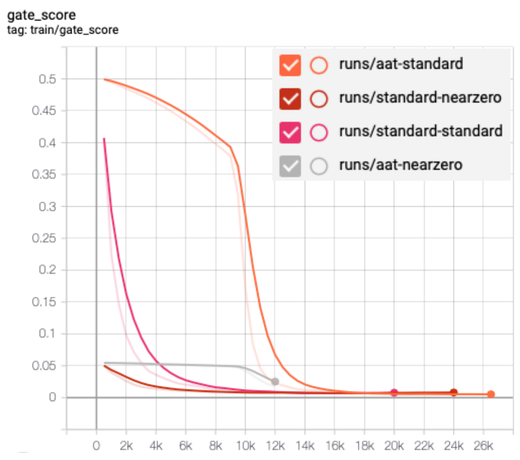
Model	Number of epochs	Training strategy	Batch size	Inference net hidden dim	NF type	Number of NF layers	Beam size	Length penalty	Max input tokens	Max target tokens
FlowSUM (Planar)	3	standard	8	600	Planar	4	4	2.0	1024	128
FlowSUM (Radial)	3	β_C-VAE	8	600	Radial	4	4	2.0	1024	128
FlowSUM (Sylvester)	3	β_C-VAE	8	600	Sylvester	4	4	2.0	1024	128
FlowSUM (RealNVP)	3	standard	8	600	RealNVP	4	4	2.0	1024	128
FlowSUM (IAF)	3	1/3 CAAT[b]	8	600	IAF	6	4	2.0	1024	128
FlowSUM (RLNSF)	3	β_C-VAE	8	600	RLNSF	4	4	2.0	1024	128
FlowSUM (RQNSF)	3	1/3 CAAT	8	600	RQNSF	4	4	2.0	1024	128

[a] "-" means not applicable.
[b] 1/3 CAAT: aggressive training for 1 epoch and non-aggressive training for 2 epochs.

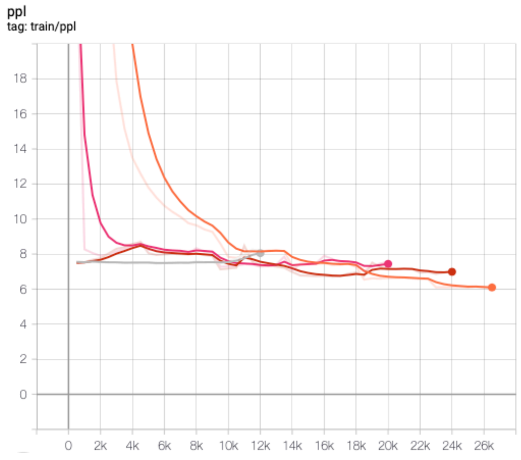
Table C.1: FlowSUM Model Hyper-parameters.

C.3 Experiments on Training Strategies and Gate Initialization

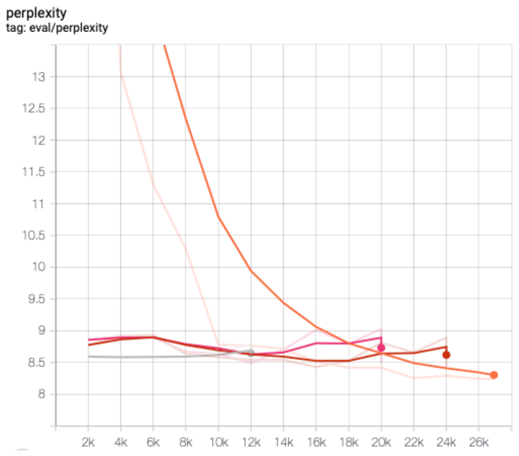
The training curves for the methods in Table 3.10 are illustrated in Figure C.1. The plot demonstrates that the gate score decreases gradually and remains high during aggressive training when CAAT is combined with standard initialization. This combination compels the model to utilize the latent code information effectively. Moreover, as presented in Figure C.1c, even though CAAT combined with standard initializa-



(a) Gate Score



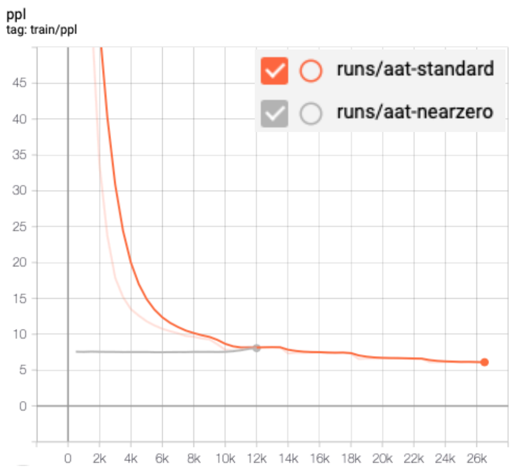
(b) Training Perplexity



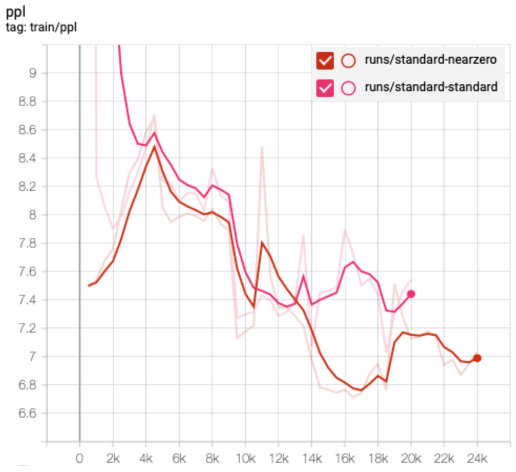
(c) Evaluation Perplexity

Figure C.1: Comparison of training strategies and gate initialization.

tion starts with a high perplexity, it achieves a lower perplexity level than other approaches by the end. By examining the training procedure in detail, Figure C.2 further indicates that CAAT contributes to greater training stability than standard training.



(a) CAAT



(b) Standard Training

Figure C.2: A closer look at the training process: CAAT vs. Standard Training.

C.4 Visualization of Latent Distribution

To gain a better understanding of how normalizing flows contribute to knowledge distillation, we selected several examples from the CNN/Daily Mail and XSum datasets and visualized the resulting latent distribution generated by the FlowSUM-PLKD model, as shown in Figure C.3 and Figure C.4. For both cases, the transformed latent code z_K exhibited a highly flexible distribution. Notably, in the CNN/Daily Mail example, the first dimension of the second example demonstrated a clear bi-modal distribution, indicating the model's ability to capture information from multiple sources. Similarly, in the XSum dataset examples, we observed distinct multi-modal patterns.

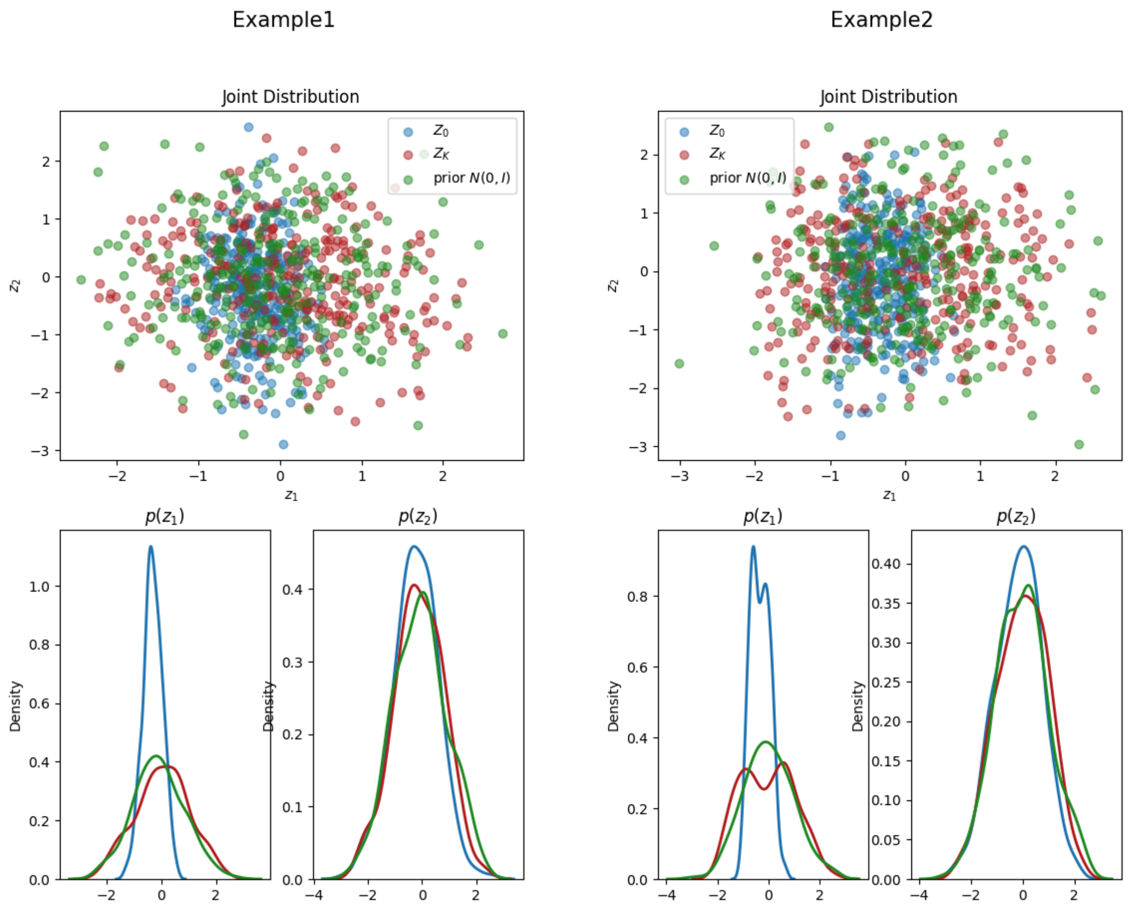
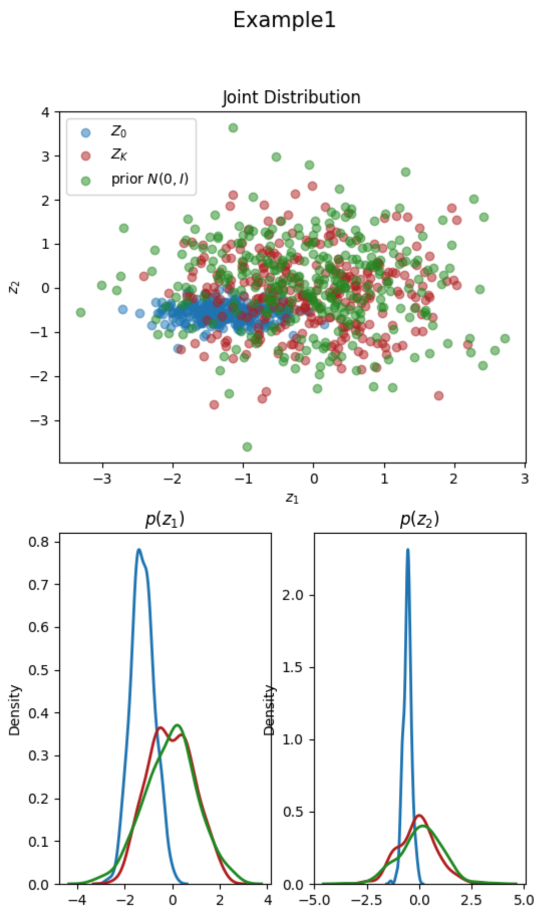


Figure C.3: Visualization of the first two dimensions of z_0, z_K, and $N(0, I)$ by FlowSUM-PLKD on CNN/DM. The right sub-figure demonstrates a clear bi-modality.



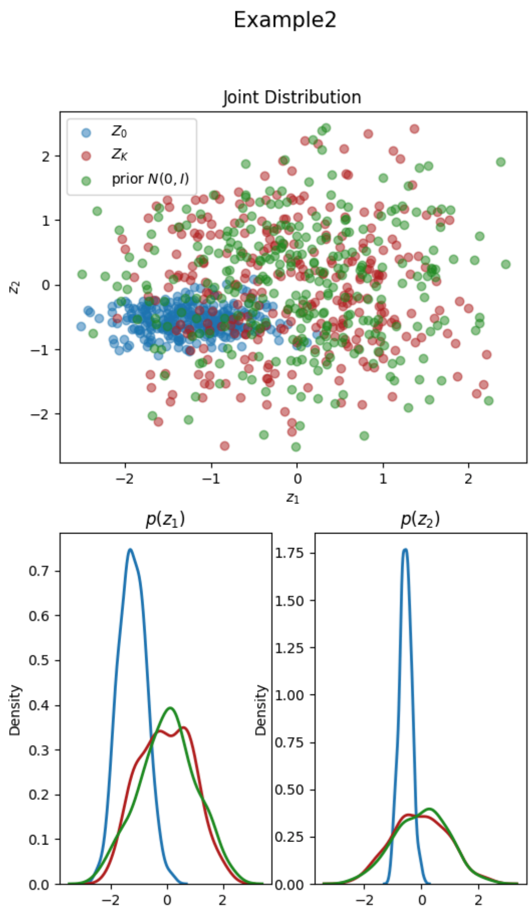


Figure C.4: Visualization of the first two dimensions of z_0, z_K, and $N(0, I)$ by FlowSUM-PLKD on XSum. Both sub-figures demonstrate distinct multi-modal patterns.

C.5 Normalizing Flows

Planar flow Proposed by Rezende and Mohamed (2015), the planar flow can be expressed as in Equation (C.3). It applies contractions or expansions in the direction perpendicular to the hyperplane $\mathbf{w}^\top \mathbf{z} + b = 0$. Its Jacobian determinant can be computed in time $\mathcal{O}(D)$ as in Equation (C.4), using the matrix determinant lemma. In addition, we need to note that this flow is not invertible for all values of $\mathbf{u}$ and $\mathbf{w}$. When the derivative of the activation function $h'(\cdot)$ is positive and bounded from above, $\mathbf{w}^\top \mathbf{u} > -\frac{1}{\sup_x h'(x)}$ is sufficient to ensure invertibility[1].

$$f(\mathbf{z}) = \mathbf{z} + \mathbf{u} h\left(\mathbf{w}^\top \mathbf{z} + b\right), \tag{C.3}$$

$$\det J = 1 + h'\left(\mathbf{w}^\top \mathbf{z} + b\right) \mathbf{w}^\top \mathbf{u} \tag{C.4}$$

where $\{\mathbf{u}, \mathbf{w} \in \mathbb{R}^D, b \in \mathbb{R}\}$ are free parameters and $h(\cdot)$ is a smooth element-wise non-linear activation function with derivative $h'(\cdot)$.

Radial flow The radial flow (Tabak and Turner, 2013; Rezende and Mohamed, 2015) takes the form of Equation (C.5). It applies radial contractions and expansions around a reference point. Similar to the planar flow, we can apply the matrix determinant lemma to calculate the Jacobian determinant in $\mathcal{O}(D)$ time, as in Equation (C.6). To guarantee invertibility, we usually require $\beta > -\alpha$[2].

$$f(\mathbf{z}) = \mathbf{z} + \beta h(\alpha, r)\left(\mathbf{z} - \mathbf{z}_0\right), \tag{C.5}$$

$$\det J = \left(1 + \frac{\alpha\beta}{h^2}\right)(1 + \beta h)^{D-1} \tag{C.6}$$

where $\mathbf{z}_0 \in \mathbb{R}^D$ is the reference point, $\beta \in \mathbb{R}, \alpha \in \mathbb{R}^+$ are free parameters, $r = \|z - z_0\|$

[1]In our code, we perform a transformation on $\mathbf{u}$: $\mathbf{u} \leftarrow \mathbf{u} + \left[\log\left(1 + \exp\left(\mathbf{w}^\top \mathbf{u}\right)\right) - 1 - \mathbf{w}^\top \mathbf{u}\right] \cdot \frac{\mathbf{w}}{\mathbf{w}^\top \mathbf{w}}$ and restrict the activation $h(\cdot)$ to be one of leakyrelu, relu, and tanh to meet this condition.

[2]In our code, we perform a transformation on $\beta : \beta \leftarrow -\alpha + \log\left(1 + e^\beta\right)$ to guarantee invertibility.

is the norm of $z - z_0$, and $h(\alpha, r) = \frac{1}{\alpha + r}$.

Sylvester flow The Sylvester flows (van den Berg et al., 2018) generalize the planar flows to have M hidden units, as in Equation (C.7). To achieve better computational efficiency, van den Berg et al. (2018) proposes the parameterization as in Equation (C.8), with which the Jacobian determinnant reduces to Equation (C.9) and can be computed in $\mathcal{O}(M)$. Similar to the planar flows, when $h'(\cdot)$ is positive and bounded from above, $\tilde{\mathbf{R}}_{ii}\mathbf{R}_{ii} > -\frac{1}{\sup_x h'(x)}$ for all $i \in \{1, \ldots, D\}$ is sufficient to ensure invertibility.

$$f(\mathbf{z}) = \mathbf{z} + \mathbf{U}h\left(\mathbf{W}^\top \mathbf{z} + \mathbf{b}\right), \tag{C.7}$$

where $\{\mathbf{U} \in \mathbb{R}^{D\times M}, \mathbf{W} \in \mathbb{R}^{D\times M}, \mathbf{b} \in \mathbb{R}^M\}$ are the free parameters and $h(\cdot)$ is an element-wise activation function.

$$f(\mathbf{z}) = \mathbf{z} + \mathbf{Q}\mathbf{R}h\left(\tilde{\mathbf{R}}\mathbf{Q}^T\mathbf{z} + \mathbf{b}\right), \tag{C.8}$$

$$\det J = \det\left(\mathbf{I}_M + \operatorname{diag}\left(h'\left(\tilde{\mathbf{R}}\mathbf{Q}^T\mathbf{z} + \mathbf{b}\right)\right)\tilde{\mathbf{R}}\mathbf{R}\right) \tag{C.9}$$

where $\mathbf{R}$ and $\tilde{\mathbf{R}}$ are upper triangular $M \times M$ matrices, and $\mathbf{Q} = (\mathrm{q}_1 \ldots \mathrm{q}_M)$ consists of an orthonormal set of vectors.

Autoregressive Flows The masked autoregressive flow (MAF) (Papamakarios et al., 2017) was motivated by MADE (Germain et al., 2015), which is an autoregressive model for density estimation. MAF generalizes the conditional distribution to be Gaussian and generates data in a recursive way as in Equation (C.10). Given a datapint $\mathbf{x}$, the inverse transformation can be performed in parallel as in Equation (C.11). The Jacobian of the inverse transformation is lower-triangular by design due to the autoregressive structure, hence its absolute determinant can be expressed as in Equation (C.12). The set of functions $\{f_{\mu_i}, f_{\alpha_i}\}$ are autoregressive neural networks following the approaches in MADE.

$$x_i = u_i \exp \alpha_i + \mu_i, \tag{C.10}$$

where $\mu_i = f_{\mu_i}\left(\mathbf{x}_{1:i-1}\right), \alpha_i = f_{\alpha_i}\left(\mathbf{x}_{1:i-1}\right)$ and $u_i \sim \mathcal{N}(0,1)$.

$$u_i = \left(x_i - \mu_i\right) \exp\left(-\alpha_i\right) \tag{C.11}$$

$$\left|\det J^{-1}\right| = \exp\left(-\sum_i \alpha_i\right) \tag{C.12}$$

Likewise, the inverse autoregressive flow (IAF) (Kingma et al., 2016) uses MADE with Gaussian conditionals and generates data as in Equation (C.13). Its Jacobian determinant has a simple form as in Equation (C.14). The main difference between IAF and MAF lies in the history variables. MAF uses previous data variables $\mathbf{x}_{1:i-1}$ to compute μ_i and α_i, whereas IAF uses previous random variables $\mathbf{u}_{1:i-1}$ for the computation. In terms of sampling and density evaluation, IAF can sample in parallel and need to evaluate sequentially, whereas MAF have to sample sequentially and can evaluate in parallel. Since in variational inference we care more about the sampling efficiency, we choose IAF in the modeling.

$$x_i = u_i \exp \alpha_i + \mu_i, \tag{C.13}$$

$$\left|\det J\right| = \exp\left(\sum_i \alpha_i\right), \tag{C.14}$$

where $\mu_i = f_{\mu_i}\left(\mathbf{u}_{1:i-1}\right)$ and $\alpha_i = f_{\alpha_i}\left(\mathbf{u}_{1:i-1}\right)$.

Affine Coupling The affine coupling layer, proposed in NICE (Dinh et al., 2015) and later generalized in RealNVP (Dinh et al., 2017) takes the following form.

$$\begin{cases} y_{1:d} & = x_{1:d} \\ y_{d+1:D} & = x_{d+1:D} \odot \exp\left(s\left(x_{1:d}\right)\right) + t\left(x_{1:d}\right) \end{cases} \tag{C.15}$$

where $s : R^d \mapsto R^{D-d}$ and $t : R^d \mapsto R^{D-d}$ are scale and translation transformation function respectively, and $\odot$ is the element-wise product.

Its Jacobian determinant can be efficiently computed as $\det J = \exp\left[\sum_j s\left(x_{1:d}\right)_j\right]$. Since the computation does not involve the Jacobian of s or t, we can make these two functions arbitrarily complex and use neural networks to model them. The coupling layers are usually composed with permutation layers to ensure every component gets modified, and since the Jacobian determinant of permutation is 1, the Jacobian determinant remains tractable.

Spline Coupling Neural spline flows (Durkan et al., 2019; Dolatabadi et al., 2020) use monotonic rational-quadratic splines or monotonic rational-linear splines as the coupling transformation to achieve more flexibility and yet remain differentiable and invertible. The monotonic rational-quadratic spline uses $K+1$ monotonically increasing knots $\left\{\left(x^{(k)}, y^{(k)}\right)\right\}_{k=0}^{K}$ to set up K bins, each of which is defined as a rational-quadratic function[3] that is monotonically increasing. It maps $[-B, B]$ to $[-B, B]$ and defines the transformation outside the range to be identity transformation. Let $s_k = \left(y^{k+1} - y^k\right) / \left(x^{k+1} - x^k\right)$ and $\xi(x) = \left(x - x^k\right) / \left(x^{k+1} - x^k\right)$, the rational-quadratic function in the kth bin takes the form of Equation (C.16) and the Jacobian determinant of the rational-quadratic neural spline flows (RQNSF) can be written as in Equation (C.17).

$$\frac{\alpha^{(k)}(\xi)}{\beta^{(k)}(\xi)} = y^{(k)} + \frac{\left(y^{(k+1)} - y^{(k)}\right)\left[s^{(k)}\xi^2 + \delta^{(k)}\xi(1-\xi)\right]}{s^{(k)} + \left[\delta^{(k+1)} + \delta^{(k)} - 2s^{(k)}\right]\xi(1-\xi)} \tag{C.16}$$

[3]A rational-quadratic function is defined as the quotient of two quadratic polynomial functions.

$$\det J = \prod_k \frac{\mathrm{d}}{\mathrm{d}x}\left[\frac{\alpha^{(k)}(\xi)}{\beta^{(k)}(\xi)}\right] = \prod_k \frac{\left(s^{(k)}\right)^2\left[\delta^{(k+1)}\xi^2 + 2s^{(k)}\xi(1-\xi) + \delta^{(k)}(1-\xi)^2\right]}{\left[s^{(k)} + \left[\delta^{(k+1)} + \delta^{(k)} - 2s^{(k)}\right]\xi(1-\xi)\right]^2} \tag{C.17}$$

The rational-linear neural splines flows (RLNSF) works similarly, except with monotonically increasing linear rational functions in each bin. Neural splines combines the best of autoregressive flows and coupling layers (such as NICE and RealNVP) in that it has both an analytic single-pass inverse and sufficient flexibility, as demonstrated in Durkan et al. (2019).

C.6 Example Analysis

In this section, we analyze several instances from CNN/Daily Mail and XSum, showcasing diverse outcomes when comparing different summarization models.

Original Text (truncated): It looks like an ordinary forest, with moss climbing up the walls and brown leaves covering the floor. But if you look closely, you will see that this picture is not all it seems. For the peaceful scene actually features a carefully painted female model. The amazing illusion is the work of German body-painting artist Joerg Duesterwald, who spent hours painting his model so she would blend in with her surroundings. The stunning set of pictures was taken in a forest in Langenfeld, Germany, yesterday. Mr Duesterwald has been painting for more than 20 years.

Gold Summary: The illusion is the work of German body-painting artist Joerg Duesterwald, who spent hours painting his model. Stunning set of pictures was taken in front of a rockface in a forest in Langenfeld, Germany, yesterday.

BART: Stunning set of images was taken in a forest near Langenfeld, Germany, yesterday by body-painting artist Joerg Duesterwald. It looks like an ordinary forest, with moss climbing up the walls and brown leaves covering the floor. But, if you look closely, you will see that this picture is not all it seems. For the peaceful scene actually features a carefully painted female model.

VAESUM: The stunning set of pictures was taken in a forest in Langenfeld, Germany, yesterday. It looks like an ordinary forest, with moss climbing up the walls and brown leaves covering the floor. But, if you look closely, you will see that this picture is not all it seems. For the peaceful scene actually features a carefully painted female model.

FlowSUM: Amazing illusion is the work of German body-painting artist Joerg Duesterwald. He spent hours painting his model so she would blend in with surroundings. Stunning set of pictures was taken in a forest in Langenfeld, Germany, yesterday.

Table C.2: Example 6766 in the CNN/Daily Mail test set: FlowSUM > BART > VAESUM. The text are colored to indicate different models. Orange means captured by Gold Summary, blue means captured by BART, red means captured by VAESUM, and purple means captured by FlowSUM.

Original Text (truncated): UFC light heavyweight champion Jon Jones ran from a crash that hospitalised a pregnant woman - but quickly came back to grab 'a large handful of cash' from the car, witnesses told police. According to police, the accident occurred in southeastern Albuquerque just before noon on Sunday local time when the driver of a rented SUV jumped a red light. The driver, whom an off-duty officer identified as Jones, ran from the scene but then returned for the cash before fleeing again, police said. 'Witnesses stated he shoved the cash into his pants and ran north jumping the fence,' the report said. Officers found a pipe with marijuana in the vehicle as well as MMA and rental car documents in Jones' name, according to the police report. Police were searching for UFC champion Jon Jones in connection with a hit-and-run accident. Albuquerque police were seeking an arrest warrant for Jones on Monday. They said he would likely face a felony charge of leaving the scene of an accident since the woman broke her arm in the crash. Police said in a news release they'd been unable to reach Jones or his lawyer. However, Jones handed himself in later the same day, with TMZ reporting he was being held at Bernalillo County Metropolitan Detention Center. ⋯ Although he is widely considered the world's best pound-for-pound mixed martial artist, Jones has endured legal problems and questionable behaviour as champion.
Gold Summary: UFC light heavyweight champion Jon Jones ran from a crash that hospitalised a pregnant woman, witnesses told police. According to police, the accident occurred in Albuquerque just before noon on Sunday when the driver of a rented SUV jumped a red light. The driver, whom an off-duty officer identified as Jones, ran from the scene but then returned for the cash before fleeing again, police said. Jones is widely considered the best pound-for-pound mixed martial artist.
BART: Albuquerque police were seeking an arrest warrant for Jones on Monday. They said he would likely face a felony charge of leaving the scene of an accident since the woman broke her arm in the crash. However, Jones handed himself in later the same day, withTMZ reporting he was being held at Bernalillo County Metropolitan Detention Center.
VAESUM: UFC light heavyweight champion Jon Jones ran from a crash that hospitalised a pregnant woman. Witnesses said he returned for 'a large handful of cash' from the car. Albuquerque police were seeking an arrest warrant for Jones on Monday. They said he would likely face a felony charge of leaving the scene of an accident since the woman broke her arm in the crash. Jones handed himself in later the same day.
FlowSUM: UFC light heavyweight champion Jon Jones ran from a crash that hospitalised a pregnant woman. Witnesses said he came back to grab 'a large handful of cash' from the car, witnesses told police. The driver, whom an off-duty officer identified as Jones, ran from the scene but then returned for the cash before fleeing again, police said. Officers found a pipe with marijuana in the vehicle as well as MMA and rental car documents in Jones' name, according to the police report.

Table C.3: Example 4627 in the CNN/Daily Mail test set: FlowSUM > VAESUM > BART. The text are colored to indicate different models. Orange means captured by Gold Summary, blue means captured by BART, red means captured by VAESUM, and purple means captured by FlowSUM.

Original Text (truncated): ... An Icelandic duo has created a snack that is made using cricket flour. Called the Jungle Bar it also contains dates, sesame seeds and chocolate. Cricket flour is said to be a good source of protein and other nutrients. The duo hopes it will encourage people in the West to eat more insects. The Jungle Bar is being developed by Icelandic duo Búi Bjarmar Aðalsteinsson and Stefán Atli Thoroddsen through their company Crowbar Protein. On Kickstarter they are seeking £10,000 ($15,000) for the insect-powered protein bar. They previously rose to fame with their Fly Factory, a micro-factory that used larvae to create foods including chocolate cake and pâté. Ingredients. Dates, sesame, sunflower and pumpkin seeds, chocolate and cricket flour. Nutrition information. The bar is 50 grams (1.7 ounces), 200 calories, contains 8 grams of high quality protein (16 per cent of the bar) and has a shelf life of 1 year. The duo say that insects are a largely untapped source of nutrients in the Western world, and they hope their product could spark a change in diet habits. To make the bar, the team has cricket flour sent in, from farm-raised crickets that have been ground down. It is then mixed with the other ingredients to make the unusual bar.

Gold Summary: Icelandic duo has created a snack that is made using cricket flour. Called the Jungle Bar it also contains dates, sesame seeds and chocolate. Cricket flour is said to be a good source of protein and other nutrients. The duo hopes it will encourage people in the West to eat more insects.

BART: An Icelandic duo has created a snack that is made using cricket flour. Called the Jungle Bar it also contains dates, sesame seeds and chocolate. Cricket flour is said to be a good source of protein and other nutrients. The duo hopes it will encourage people in the West to eat more insects. On Kickstarter they are seeking £10,000 ($15,000) for the insect-powered bar.

VAESUM: The Jungle Bar is being developed by Icelandic duo Búi Bjarmar Aðalsteinsson and Stefán Atli Thoroddsen through their company Crowbar Protein. Cricket flour is said to be a good source of protein and other nutrients. The duo hopes it will encourage people in the West to eat more insects. To make the bar, the team has cricket flour sent in, from farm-raised crickets that have been ground down.

FlowSUM: An Icelandic duo has created a snack that is made using cricket flour. Called the Jungle Bar it also contains dates, sesame seeds and chocolate. The Jungle Bar is being developed by Búi Bjarmar Aðalsteinsson and Stefán Atli Thoroddsen through their company Crowbar Protein. On Kickstarter they are seeking £10,000 ($15,000) for the insect-powered protein bar. They previously rose to fame with their Fly Factory, a micro-factory that used larvae to create foods including chocolate cake and pâté.

Table C.4: Example 4120 in the CNN/Daily Mail test set: BART > VAESUM > FlowSUM. The text are colored to indicate different models. The text are colored to indicate different models. Orange means captured by Gold Summary, blue means captured by BART, red means captured by VAESUM, and purple means captured by FlowSUM.

Original Text (truncated): The Office for National Statistics said industrial output fell 0.7% compared with January, when it dropped 0.3%. Unexpectedly warm weather drove the change, because it led to a fall in electricity and gas demand, the ONS said. Construction output fell by 1.7% in February, down from a revised January reading of zero growth. The construction figure, the biggest drop in nearly a year, was mainly the result of a 2.6% fall in the housebuilding sector. Meanwhile, the UK's deficit in goods and services widened to Â£3.7bn in February, from a revised figure of Â£3bn in January. According to the ONS, the deficit was fuelled by what it called "erratic items", such as imports of gold and aircraft. "The overall trade deficit worsened, but excluding erratic items, the picture improved, as imports fell more than exports," said ONS senior statistician Kate Davies. Howard Archer, chief UK and European economist at IHS Markit, called the figures "a disappointing package of data for the UK economy which fuels suspicion that GDP growth slowed markedly, largely due to consumers becoming more cautious". He added: "We suspect UK GDP growth in the first quarter of 2017 slowed to 0.4% quarter-on-quarter from 0.7% quarter-on-quarter in the fourth quarter of 2016 - this would be the weakest growth rate since the first quarter of 2016."

Gold Summary: Activity in the UK's industrial and construction sectors shrank in February, new figures show.

BART: UK industrial output fell for the second month in a row in February, official figures have shown.

VAESUM: Industrial output in the UK fell for the second month in a row in February, official figures have shown.

FlowSUM: Activity in the UK's industrial and construction sectors shrank in February, according to official figures.

Table C.5: Example 2924 in the XSum test set: FlowSUM > BART > VAESUM.

Original Text (truncated): In December, the government announced finalised plans for a cull, initially in pilot areas, as a way to curb the spread of tuberculosis in cattle. In applying for judicial review, the Badger Trust says culling will not stop TB and may in fact help spread it. Other campaign groups are considering action under the Bern Convention, which protects European wildlife. The government's plans are likely to result in farmers funding contractors to shoot badgers in a number of areas of England, with two initial pilots in west Gloucestershire and west Somerset taking place later this year. "We have identified some serious flaws in the way by which the Secretary of State [Caroline Spelman] reached her decision to cull badgers," said Gwendolen Morgan of Bindmans solicitors, lawyer for the Badger Trust. "Given that Defra's proposals come at an enormous cost to farmers, and threaten to prompt rather than prevent the spread of disease, we hope that this ill-conceived decision will be struck down by the court." She pointed to government projections that culling would reduce TB incidence by 12-16% over nine years.
Gold Summary: The Badger Trust has launched a new legal challenge to the government's plans to cull badgers in England.
BART: The Badger Trust has launched a legal challenge to the government's plans to cull badgers in England.
VAESUM: The Badger Trust is taking legal action against the Department for Environment, Food and Rural Affairs (Defra) over plans to cull badgers in England.
FlowSUM: The Badger Trust has launched a legal challenge to the UK government's plans to cull badgers in England and Wales.

Table C.6: Example 5737 in the XSum test set: BART > FlowSUM > VAESUM.

Original Text (truncated): The response from many in that time has been: "Let's get on with it." That view was shared by the First Minister Carwyn Jones until recently when he altered his opinion and said that we should only start the official Brexit negotiations in the early part of next year. My sense is that the public will be flexible on the timing up to a point, as long as they are given a clear sense of direction. The majority of the political establishment have had to come to terms with the fact that most people ignored their advice to remain. So much for being in touch with the electorate. In conversations with politicians on the remain side since, I have come across a mix of bewilderment, frustration and sadness. And while people like me spend a lot of time talking and writing about a Welsh political dynamic, on this subject at least, Wales was a carbon copy of England. In stark contrast, those that supported leaving feel vindicated by their campaign, and now believe they are the ones in touch with vast swathes of the population. The referendum result was a devastating indictment of the effectiveness of the billions of pounds of EU funds spent trying to regenerate economically deprived communities. The brutal reality is that those who were most likely to vote to leave lived in communities where most EU money had been spent. It is an extraordinary paradox that raised eyebrows far further afield than Wales.

Gold Summary: It has been a month since Wales voted to leave the European Union.

BART: It has been more than a year since the UK voted to leave the European Union.

VAESUM: It has been a year since the EU referendum result, and in that time I have spent a great deal of time talking to politicians on both sides of the political spectrum about what they think about Brexit.

FlowSUM: Since the referendum result on 23 June, I have spent a lot of time talking about the implications for Wales and the Welsh political establishment.

Table C.7: Example 9512 in the XSum test set: BART > VAESUM > FlowSUM.

Appendix D

Chapter 4 Appendices

D.1 Data Preprocessing

The measurement and sensor data require basic preprocessing, such as removing missing, duplicate, and uninformative records. For the measurement data, the abnormal events at every step are a long list of strings, so we extract the abnormalities by textual processing and then transform them into binary variables, with one indicating abnormal. And for the sensor data, since they are collected every 3 seconds, and most values do not change much within a step, we aggregate them by averaging. We use the mean instead of the median because the latter is robust against outliers and insensitive to abnormalities.

D.2 Constraints by Domain Knowledge

1. For a node from 'MEAS(2)_MON', its parent must come from the same stage, and if its child node is also from 'MEAS(2)_MON', then the steps in between must be 'MEAS*' or 'INSPECT'.

2. Sensors from different tools cannot be linked.

3. If a child node does not come from 'MEAS*', 'INSPECT', or 'TEST*', then

either its stage is ‘*_VE’ or its step is one of [‘DRYETCH’, ‘LAP_EBARA’, ‘ASH’].

4. Two nodes that are 60 steps apart cannot be linked.

5. The child node cannot be from ‘INSPECT’.