# Deep learning with constraints for answer-agnostic question generation in legal text understanding

by

Deepti Lamba

B.Tech., Mody University, 2008

M.Tech., Delhi University, 2011

---

AN ABSTRACT OF A DISSERTATION

submitted in partial fulfillment of the
requirements for the degree

DOCTOR OF PHILOSOPHY

Department of Computer Science
Carl R. Ice College of Engineering

KANSAS STATE UNIVERSITY
Manhattan, Kansas

2021

# Abstract

The aim of this dissertation is to develop constraint-based methods that extend and improve on current deep learning neural networks such as transformers and sequence-to-sequence (seq2seq) models, for the problem of question generation based on the analysis of the text of legal agreements, particularly privacy policies.

A privacy policy is a legally binding agreement between a customer and service provider.

This dissertation focuses on analyzing a privacy policy document to generate questions that capture entities and the relationships between them. Another area of focus is the generation of constraints based on domain knowledge and their application to the deep learning network during the question generation process. A possible use case of this research is development of test corpus for question answering systems in the privacy domain because the shortage of sufficiently large corpora poses a key challenge in the development of question answering and question generation systems.

Question generation is the task of generating an interrogative sentence based on some text. Current approaches to question generation use sequence-to-sequence models with additional information like answers, positions of the answers, part-of-speech details, named entity tags among others. The idea behind such approaches is that these models can benefit from additional information about the text (i.e., sentence or paragraph).

Recently, transformer-based approaches that offer the benefit of attention mechanism have also been used for generating questions. Transformers have achieved state-of-the-art results in many natural language processing tasks including text classification, machine translation, language understanding, co-reference resolution, and summarization.

However, the contribution of transformers towards a task like question generation has not been as significant.

This research tries to find ways of improving existing approaches by injecting domain knowledge, modeled as a combination of logical and linguistic constraints, into these deep learning models during the training and validation phases. This work also explores design and implementation of different kind of constraints that can better direct the deep learning model towards the expected output, which in this case refers to syntactically and semantically correct and relevant questions. Another contribution of this research is the creation of custom labels for named entities in the privacy policy domain. Results show that adding some form of domain specific constraints improves the performance of the aforementioned models as compared to the performance of state-of-the-art models on the test bed used in this work. For the given test bed, constrained seq-to-seq approaches perform better than the constrained transformer-based approach.

# Deep learning with constraints for answer-agnostic question generation in legal text understanding

by

Deepti Lamba

B.Tech., Mody University, 2008

M.Tech., Delhi University, 2011

---

A DISSERTATION

submitted in partial fulfillment of the
requirements for the degree

DOCTOR OF PHILOSOPHY

Department of Computer Science
Carl R. Ice College of Engineering

KANSAS STATE UNIVERSITY
Manhattan, Kansas

2021

Approved by:

Major Professor
Dr. William H. Hsu

# Copyright

# Abstract

The aim of this dissertation is to develop constraint-based methods that extend and improve on current deep learning neural networks such as transformers and sequence-to-sequence (seq2seq) models, for the problem of question generation based on the analysis of the text of legal agreements, particularly privacy policies.

A privacy policy is a legally binding agreement between a customer and service provider.

This dissertation focuses on analyzing a privacy policy document to generate questions that capture entities and the relationships between them. Another area of focus is the generation of constraints based on domain knowledge and their application to the deep learning network during the question generation process. A possible use case of this research is development of test corpus for question answering systems in the privacy domain because the shortage of sufficiently large corpora poses a key challenge in the development of question answering and question generation systems.

Question generation is the task of generating an interrogative sentence based on some text. Current approaches to question generation use sequence-to-sequence models with additional information like answers, positions of the answers, part-of-speech details, named entity tags among others. The idea behind such approaches is that these models can benefit from additional information about the text (i.e., sentence or paragraph).

Recently, transformer-based approaches that offer the benefit of attention mechanism have also been used for generating questions. Transformers have achieved state-of-the-art results in many natural language processing tasks including text classification, machine translation, language understanding, co-reference resolution, and summarization.

However, the contribution of transformers towards a task like question generation has not been as significant.

This research tries to find ways of improving existing approaches by injecting domain knowledge, modeled as a combination of logical and linguistic constraints, into these deep learning models during the training and validation phases. This work also explores design and implementation of different kind of constraints that can better direct the deep learning model towards the expected output, which in this case refers to syntactically and semantically correct and relevant questions. Another contribution of this research is the creation of custom labels for named entities in the privacy policy domain. Results show that adding some form of domain specific constraints improves the performance of the aforementioned models as compared to the performance of state-of-the-art models on the test bed used in this work. For the given test bed, constrained seq-to-seq approaches perform better than the constrained transformer-based approach.

# Table of Contents

# List of Figures

# List of Tables

# Acknowledgments

I would like to thank all those who have supported me during my doctoral program here at Kansas State University. First and foremost, I would like to thank my brother, Yash Lamba for all his encouragement and support. My doctorate would have never been possible without him. Next, I would like to thank my uncle, Mr. Lalit Kumar and my aunt, Ms. Sapna Kumar for being a constant source of motivation and support. I also extend my gratitude to my colleague and friend, Majed Alsadhan, for supporting me during the entire doctoral program.

I further express my gratitude to all my committee members, Dr. Torben Amtoft, Dr. Mitchell Neilsen, and Dr. Caterina Scoglio. Last, but definitely not the least, I thank my advisor, Dr. William Hsu for spending innumerable hours in refining my work and teaching me so much during my time with him. He has been an advisor and mentor in the truest sense.

# Dedication

To my late maternal grandfather.

# Preface

This dissertation comprises six chapters, all of which I have written myself. The data set used in this dissertation was published by Ahmad et al. (2020)[1]. Some of the contents of Chapter 2, sections 2.1 and 2.2, have been submitted for publication. A version of chapter 4 has also been submitted for publication. Some sentences from these publications have been used verbatim in this dissertation. Dr. William Hsu has provided a significant review for the publications. The remaining chapters have been written by me, with important suggestions from Dr. William Hsu.

# Chapter 1

# Introduction

> You cannot answer a question you cannot ask, and you cannot ask a question that you have no words for.
>
> — *Judea Pearl, The Book of Why: The New Science of Cause and Effect*

The task of question generation (QG) is defined as generating an interrogative sentence from a given paragraph or sentence (which is referred to as the context). It is considered the inverse problem of question answering (QA), where given a question and context, the task is to generate an answer. Question generation is not as well researched as question answering, but in recent years, the task has caught on with the research community. The task of question generation is also considered an important sub-task for question answering systems because to extract appropriate answers from a given context, one must ask the right question. In this work, I focus on the task of generating questions in a legal area: privacy policy understanding.

In this chapter, I provide some background on the task of question generation, followed by a formal definition of the problem statement and the purpose of my dissertation. This is followed by the primary research questions that I will answer in my work. This chapter will also detail the assumptions on which my research is based, the limitations, and the scope of my work. Thereafter, I will present key task definitions.

## 1.1 Background

The task of question generation is defined "as the automatic generation of questions (Factual questions, Yes/No questions, Why-questions, etc.) from inputs such as text, raw data, and knowledge bases" [2]. The rationale behind the task of question generation is to automatically generate grammatically and syntactically correct questions based on an understanding of the context.

Some popular use cases of question generation include the following:

- creating intelligent tutoring systems in education[3;4].
- enhancing human-machine interactions in dialogue systems like chat bots[5].
- improving the performance of question answering systems by reducing human labor needed to generate large-scale data sets[6–8].

Kurdi et al. (2020)[4] claim that the main purpose of question generation is assessment in the field of education. They also list tutoring or self-assisted learning systems and experimental settings as other applications of question generation.

## 1.2 Motivation

In this research, I focus on the problem of automatically generating questions from privacy policies of various companies. A privacy policy document is a legal document that explicitly discloses the data gathering, handling and processing policies of an organization, in other words, how an organization or website collects, processes, and handles customer/visitor/user data. This privacy policy document also indicates whether the confidentiality of gathered customer data is maintained or whether it is shared with or sold to third parties. The gathered data can include personal identifying characteristics like name, age, gender, address, phone number, email, nationality, religion, or race, among others. Websites and other applications can also gather data such as the internet protocol address (IP address) of the user, operating system specifications, browser information, cookies, activity logs, etc. Businesses

are required by law to share their privacy policies with the users of their services. Users must read these policies to protect their privacy. Unfortunately, many users agree to privacy policies without reading them. The main reason for this is the perception that privacy policy documents are long, extremely verbose, and hard to understand, and therefore reading them is too time consuming.

Many studies and surveys have been conducted over the years to see why people accept privacy policies without reading them. Obar and Oeldorf-Hirsch (2018)[9] conducted an empirical investigation of student reading behavior for privacy policy statements and terms of service (TOS) policy documents. The authors asked people to join a fictitious social networking service called NameDrop. This study demonstrates how organizations or individuals with malicious intent can gather and exploit user data without users ever knowing about it. The privacy policy document used in the study included a clause that stated, "By agreeing to the TOS, participants would give up their first-born child to NameDrop." However, even that clause did not deter the participants from agreeing to the terms of service. Results revealed that almost all participants exhibited two kinds of behavior: they either ignored the policy document altogether or paid insufficient attention to the policies. Many other studies in the past have revealed similar user behavior[10–14].

A frequently asked question (FAQ) section in privacy policy documents can help alleviate the problem by revealing some important topics mentioned in privacy policies. However, not all services offer an FAQ section and even when present, it may not contain many relevant questions. Thus, a question answering system is necessary, making it easier for users to understand privacy policy documents by helping them obtain answers to specific questions without spending the time needed to read the entire document. Considering this, I will be developing an automatic question generation system as a first step towards building a full-fledged question answering system. By generating a better class of questions, the question generation system can improve the quality of any question answering system for privacy policy documents, which in turn can help end users better absorb the privacy policies and make well-informed decisions.

## 1.3 Problem Statement

I aim to create a system for question generation that takes as input a paragraph of text, and outputs a question based on the text. The system should perform this task without using answer sequence as input to the model.

We formally define the problem of question generation as the following: Given a passage (or context from a policy document), $X_p = (x_1, x_2, ..., x_n)$ as input, the model aims to generate a question, Y $= (y_1, y_2, ..., y_T)$. The goal is to find the best $\bar{Y}$:

$$\bar{Y} = \underset{x}{\text{argmax}} P(Y|X_p)$$

where $P(Y|X_p)$ is the conditional log-likelihood of the predicted question sequence y, given the input x.

## 1.4 Purpose of Study

The novel contributions of this study include the following:

1. **Studying the privacy policy domain for building a question generation system:**

   A literature review suggested that research in this area has focused on extracting knowledge using rule-based systems, topic modeling-based systems, and question answering systems. To date, no effort has been made to generate questions in this domain.

2. **Creating custom named entities for the privacy policy domain:**

   A study of the data in the domain suggested that standard named entity recognition (NER) tools fail to provide labels for more than half the data set being used, entirely due to the nature of privacy policy documents. This work also focuses on using the custom named entities to improve the question generation model.

3. **Using constraints to inject domain knowledge into deep learning models for the task of question generation:**

To date, constraints have not yet been used for the question generation task. The key idea is to use constraints to inject domain specific knowledge into the deep learning model. The focus will be on creating constraints that can be generalized to other domains.

4. **Evaluating the generated questions with standard, automatic evaluation metrics and studying the effectiveness of these evaluation metrics:**
   The idea of this analysis is to gauge the need for a better evaluation scheme for generated text when there is no gold standard for comparison.

## 1.5 Primary Research Questions

The primary research questions that this work aims to answer are listed below:

- How can existing state-of-the-art question generation models be enriched by including domain knowledge to better produce syntactically and semantically correct questions that are more diverse?
- What kind of constraints can better inject prior-knowledge into the model?
- Does transfer learning alone help improve the performance of question generation systems using a small data set?

## 1.6 Assumptions and Limitations

Question generation systems, like the one discussed in this document, assume that each paragraph of the source text contains a question-worthy concept and can be used to generate at least one question.

The system falls into the category of "answer-unaware" question generation systems because it does not consider the answer position or text. Such systems suffer from random question generation; given a source text, several questions can be inferred depending on the

perspective. For example, for the given text: "Josh, who is a salesman is traveling to New York City", the following questions can be generated:

- Who is travelling to New York City?
- Where is Josh going?
- Who is Josh?
- Which salesman is traveling to New York City?

Making the system "answer-unaware" is necessary because of the privacy policy data set used for this research. Some questions in the data set could not be directly answered from the given text. For example, the following paragraph comes from the privacy policy of TGI Fridays, with the answer marked in bold in the source text:

**Source:** "The information that you provide is collected by TGI Fridays. In the case of links to our gift card and guest recognition sites, the information you voluntarily provide at those sites will only be shared with those service vendors who help TGI Fridays administer those websites or mobile application and the services they provide. In any case, TGI Fridays is the lawful "owner" of the information and each of these vendors may use the information only for the purpose of administering the digital or mobile application and its services for TGI Fridays, and will take all necessary precautions to protect the information. Ownership of any information you provide us will be held solely by TGI Fridays. We **will not** sell ownership of this data to any other company or organization."

**Question:** Does the third party follow the privacy practice?

**Answer:** will not

It is evident that the question is not explicitly answered by the source text. The answer can only be inferred by a human. The data consists of such question-answer pairs, that cannot be discarded due to the small size of the data set. Consequently, answers have not been considered for the question generation task.

## 1.7 Key Definitions

The following are key terms and definitions that will be used in this dissertation:

- *Question Generation* is defined as the task of automatically generating questions, given a paragraph/source text and/or answers.
- *Source Text* refers to a paragraph taken from the privacy policy document that is used to generate a question.
- *Answer Phrase* refers to a span of words in the source text that contains the answer to the question.
- *Reference Question* refers to the question generated by human annotators given a specific source text.
- *Candidate Question* refers to the predicted questions that are generated by the machine learning model.

## 1.8 Overview

This dissertation presents a review of the background literature in question generation in Chapter 2. The chapter briefly covers rule-based question generation and focuses on neural question generation in detail. This is followed by a background on methodology used for incorporating domain or prior knowledge in machine learning models in Chapter 3. The results of my research are presented in Chapters 4 and 5.

Chapter 4 presents the results of applying existing sequence-to-sequence and transformer-based models to the privacy policy data set, where the context input has been augmented with named entity labels that I created for this data set. The results presented in this chapter establish a baseline for Chapter 5, which focuses on using logical and numerical constraints for the question generation task in the privacy policy domain. This chapter presents the contributions of this research to the state of the field, as well as, towards the legal domain:

understanding privacy policy documents. The results presented in this chapter pave the way for future research presented in Chapter 6, which also presents a summary of the main contributions of this work.

# Chapter 2

# Related Work on Question Generation

In this chapter, I present relevant literature on question generation. Methods used for the task of question generation can be broadly classified into two major categories: (a) rule-based approaches; and (b) deep learning-based approaches. Traditional rule-based approaches require deep linguistic knowledge to generate hand-crafted rules to transform a declarative sentence into an interrogative sentence. These systems require intensive manual labor to create rules, and those rules may not generalize to other domains. On the other hand, deep learning-based approaches provide an end-to-end solution that is driven by data as opposed to hand crafted rules.

## 2.1 Rule-based Question Generation

A typical rule-based system transforms the input sentence into its syntactic representation, which is then used to generate a question. Rule-based systems offer some benefits over neural network-based models: they are easier to interpret, they allow developers greater control over model behavior, and they typically require less data to achieve a comparable performance.

Mitkov and Ha (2003)[15] used a set of general transformational rules to generate multiple-choice tests. Gates (2008)[16] used tree manipulation rules to generate fact-based reading comprehension questions. Khullar et al. (2018)[17] used a syntax-based system that ran on

dependency parse information of the input sentence. Heilman and Smith (2009)[3;18] also included a statistical component for scoring questions to generate a ranked set of questions about text. Chali and Hasan (2015)[19] focused on the task of topic-based question generation, where given a body of text pertaining to a topic, they used rules based on named entity information and semantic role labels.

A recent work by Dhole and Manning (2020)[20] pointed out the lack of variety in questions generated by previous rule-based approaches, which have used simple syntactic transformations to create questions using declarative sentences. These two authors also discussed the lack of syntactic fluency in questions generated by neural question generation models. Their work used syntactic rules that leveraged universal dependencies, lexical information, shallow semantic parsing, and custom rules to transform declarative sentences into question/answer pairs.

## 2.2 Neural Network-based Question Generation

### 2.2.1 Sequence-to-Sequence Framework

Sequence-to-sequence framework uses an encoder to read the input text and uses a decoder to generate the question. Most existing neural question generation systems use a sequence-to-sequence framework (Sutskever et al., 2014[21]) in conjunction with the attention mechanism (Bahdanau et al., 2014)[22]. The attention mechanism helps the decoder pinpoint the most relevant part of the input text while generating the question. Neural question generation models can be broadly classified as answer-aware and answer-unaware models[23]. Answer-aware models take the context and the answer and/or the answer position as input for the encoder. This allows the system to generate targeted questions. Answer-unaware models, On the other hand, tend to generate questions without a specific target. Pan et al. (2019)[24] presented recent advances in neural question generation.

The earliest work on using neural networks for question generation is credited to Du et al. (2017)[25], who used a sequence-to-sequence model with an attention mechanism to achieve

better performance than rule-based systems. Their model used the context-question pair for training and did not use the answer. Another prominent work by Du et al. (2017)[26] was their answer-unaware model that divided the question generation task into two steps: (1) identifying question-worthy sentences; (2) using those sentences as input for generating questions using their earlier sequence-to-sequence model with attention. The task of sentence selection was performed using a neural sequence tagging model.

An example of answer-aware model was presented by Zhou et al. (2017)[27] who used the answer positions along with additional features generated from named entity recognition and part-of-speech tags. Their work also used the same attention mechanism as Du et al. (2017)[25]. However, these types of answer-aware models included words from the answer as part of the generated question, resulting in useless questions. Kim et al. (2019)[28] replaced the answer with a special token in the paragraph to prevent the predicted question from including answer words. Song et al. (2018)[29] followed a similar path, separately encoding passage (source text) and answer, and using the multi-perspective context matching algorithm[30] between the two encodings as additional input to the decoder. Both these efforts used recurrent neural network to encode the answer feature separately.

The input to sequence-to-sequence models has also been augmented with additional information to produce better questions. Hu et al. (2018)[31] provided the topic of question as additional information to the model. Their work focused on topic-specific question generation using sequence-to-sequence learning framework, which use three components: a topic encoder, an answer encoder, and a decoder. Experimental results indicated that the model performed better than conventional baseline models because of additional information on the topic of the question. Cao et al. (2020)[23] defined the question generation task as a one-to-many mapping problem and incorporated auxiliary information from data to train an attentive sequence-to-sequence model with a copying mechanism (Gulcehre et al., 2016)[32].

Another prominent direction in neural question generation research has been question word generation. In sequence-to-sequence models, the decoder performs the twin task of generating the interrogative word and the remaining text of the question. A few researchers have separated these twin tasks and improved the performance of the question generation

model. Sun et al. (2018)[33] proposed a question generation model that first generated an interrogative word using the answer type and then generated the remaining text of the question using the relative distance between context words and the answer. Similarly, Kang et al. (2019)[34] used a pipe-line system consisting of two modules: first an interrogative word classifier and second a question generator that used the interrogative word from the first step. This approach ensured that the decoder could focus entirely on the remaining question text and not on the interrogative word.

Ma et al. (2020)[35] proposed an attention-based sequence-to-sequence model, which much like the one proposed by Zhou et al. (2017)[27], took the named entity labels (Sang and De Meulder, 2003[36]), part of speech tags (Brill, 1992[37]), alphabetic case, and answer position features as input. They also used a pointer-generator network[32] and copy mechanism[38] to better exploit the answer position-aware features. Harrison and Walker (2018)[39] also used linguistic features like named entity recognition, word case, and entity co-reference resolution to build a sequence-to-sequence model with two encoders: one for token level embedding, and the other for sentence level embedding. This enabled the decoder to capture word level and sentence level meanings.

In question generation using sequence-to-sequence models, the focus remained on the ability to augment the input with additional information to generate targeted and more focused questions. In my work, I also aim to extend this approach by augmenting the basic sequence-to-sequence models with additional background information specific to the privacy policy domain. Besides the research described above, some research in question generation has made use of reinforcement learning, some has made use of knowledge-graphs, and some work has also been done on visual question generation. In the following paragraphs, I provide a brief overview of such research works.

**Question-specific Rewards** Yuan et al. (2017)[40] used a sequence-to-sequence approach conditioned on context and answer and then used REINFORCE algorithm[41] to maximize the model's expected reward. Yao et al. (2018)[42] modeled the question generation task as a one-to-many problem, where given a context and answer, multiple valid questions could be generated. They used the GAN[43] framework for the task. Chen et al. (2019)[44] pro-

posed a reinforcement learning-based graph-to-sequence (Graph2Seq) model with a novel Bidirectional Gated Graph Neural Network-based encoder. Xie et al. (2020)[45] designed three different reinforcement learning rewards for each of the following metrics associated with the generated questions: (1) fluency (whether the question is grammatical and follows the correct logic), (2) relevance (whether the question is relevant to the document), and (3) answerability (whether the question is answerable given the document). Wang et al. (2020)[46] focused on the task of deep question generation[47] with reinforcement learning and an answer-driven encoder-decoder model.

**Question Generation from Knowledge Graphs** uses knowledge graphs for generating questions. Elsahar et al. (2018)[48] produced one of the earliest prominent works in this space when they proposed a neural model that generated questions from knowledge base triples (predicates, subject, object) in a Zero Shot[49;50] setting. Hu et al. (2019)[51] used knowledge base (KB) triples and textual corpus for creating a unified framework that combines question answering and question generation with the aim of improving question answering systems. Kumar et al. (2019)[52] used a sub-graph and an answer as input to a transformer-based model to generate questions. Bi et al. (2020)[53] used a sub-graph augmented by auxiliary information acting as input to an encoder and used a constrained decoder to generate questions.

**Visual question generation** is an emerging topic that takes images as input to automatically generate questions. Prominent work in this direction includes the following: Mostafazadeh et al. (2016)[5] introduced the task of Visual Question Generation and created three data sets with a total of 75,000 questions; Zhang et al. (2016)[54] generated captions for images and generated corresponding questions conditioned on question type and a caption; Jain et al. (2017)[55] used a combination of variational auto-encoders and long-short-term-memory cells to generate a diverse set of questions for a given input image; Krishna et al. (2019)[56] used a variational auto-encoder to generate questions aimed at expecting a specific response type; and Shukla et al. (2019)[57] focused on the task of goal-oriented visual dialogue that combines information gain with reinforcement learning.

### 2.2.2 Transformer-based Question Generation

In recent years, some research has been conducted on generating questions using transformer-based[58] approaches. Some of the research in question generation that has used transformers includes the following: Matsumori et al. (2021)[59] proposed a Unified Questioner Transformer (UniQer) for visual question generation; Scialom et al. (2019)[60], used transformer for answer-agnostic question generation on SQuAD data set[61]; and Chan and Fan (2019)[62] used pre-trained BERT model[63] to generate more semantically fluent and coherent questions. BERT-based models use the input context and target answers to generate questions. This allowed the models to improve the state-of-the-art results significantly on the benchmark Stanford Question Answering Data set or SQuAD data set. Varanasi et al. (2020)[64] extended BERT-based models with copy mechanisms.

This work uses the "Text-To-Text Transfer Transformer"(T5) proposed by Raffel et al. (2020)[65] and described in section 2.3.2.

## 2.3 State-of-the-art

The best performing models on SQuAD data set discussed in the literature[25;66] use sequence-to-sequence models at their core. The sequence-to-sequence models have been improved using extra mechanisms, and/or extra input features. Very recently, Transformers[58;67] have been used to generate questions. This section provides an overview of these two networks. These approaches encode the input source text and then decode the embedded information into an output question.

### 2.3.1 Sequence-to-Sequence Models

The most common sequence-to-sequence models are encoder-decoder models, which use a recurrent neural network[68]. The recurrent neural network is trained to map an input sequence to an output sequence, which may or may not be of the same length. Cho et. al (2014b)[69] provided the simplest recurrent neural network architecture for such mapping. Sutskever

et al. (2014)[21] used this architecture for language translation and obtained state-of-the-art results. This section gives a brief overview of the encoder and decoder models. Sutskever et al. (2014)[21] used an encoder-decoder architecture where a reversed input sequence is read in its entirety and encoded to a fixed-length internal representation. This internal representation was used as input to the decoder that output words until the end of token was reached. Long short term memory networks[70] were used for both the encoder and decoder.

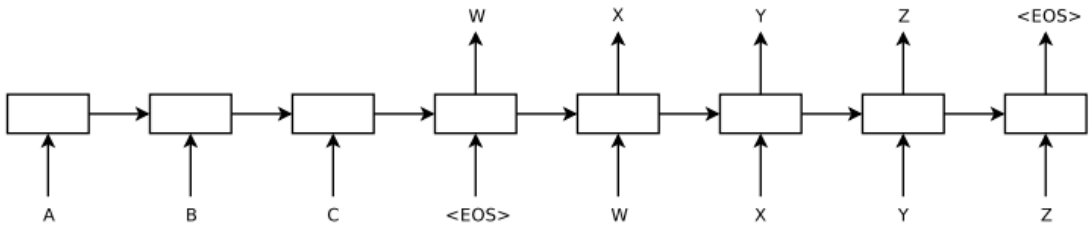


**Figure 2.1**: *Sequence-to-Sequence Model (Sutskever et al., 2014)[21]*

### Encoder

An encoder is a stack of several recurrent units (LSTM[70] or GRU[71] cells for better performance) where each unit accepts a single element of the input sequence, collects information for that element, and propagates it forward. In question generation, the input sequence is a collection of all words from the context. Each word is represented as $x_t$, where $t$ is the timestamp of that word. The hidden state at time t, $h_t$, is computed using the formula given below:

$$h_t = f(W^{hh}h_{t-1} + W^{hx}x_t) \tag{2.1}$$

where $h_{t-1}$ represents the previous hidden state, and $W$ represents the weight matrices. A **GRU encoder** calculates the hidden state $h_t$ as follows[71;72]:

$$z_t = \sigma(W_{xz}x_t + U_{hz}h_{t-1}) \tag{2.2}$$

$$r_t = \sigma(W_{xr}x_t + U_{hr}h_{t-1}) \tag{2.3}$$

$$\tilde{h}_t = tanh(W_{xh}x_t + U_{rh}(r_t \otimes h_{t-1})) \tag{2.4}$$

$$h_t = (1 - z_t) \otimes h_{t-1} + z_t \otimes \tilde{h}_t \tag{2.5}$$

where $\sigma$ is the sigmoid function, $\otimes$ is an element-wise multiplication operator, $x_t$ is the input vector, $z_t$ is the update gate, $r_t$ is the reset gate, $\tilde{h}_t$ candidate activation, and W and U are the weight matrices.

### Decoder

A decoder is a stack of several recurrent units where each unit accepts the hidden state from the previous unit to predict output $y_t$ at time step $t$ as well as its own hidden state. In question generation, the output sequence is a collection of all words from the question. Each word is represented as $y_t$, where $t$ is the position of that word in the sequence. The output $y_t$ at time step $t$ is computed using the formula given below:

$$y_t = f(W^s h_t) \tag{2.6}$$

### Attention Mechanism

The standard encoder-decoder architecture performs poorly on long input sequences because the decoder only uses the last hidden state of the encoder. Bahdanau et al. (2014)[22] extended the simple encoder-decoder structure by allowing a model to automatically search for relevant parts of a source sentence essential for predicting a target word correctly. This approach does not convert all the source text information into a fixed length vector, but rather encodes the source sentence into a sequence of vectors, and the model chooses a subset of these vectors for decoding.

Conditional probability for each:

$$p(y_i|y_1, ..., y_{i-1}, x) = g(y_{i-1}, s_i, c_i) \tag{2.7}$$

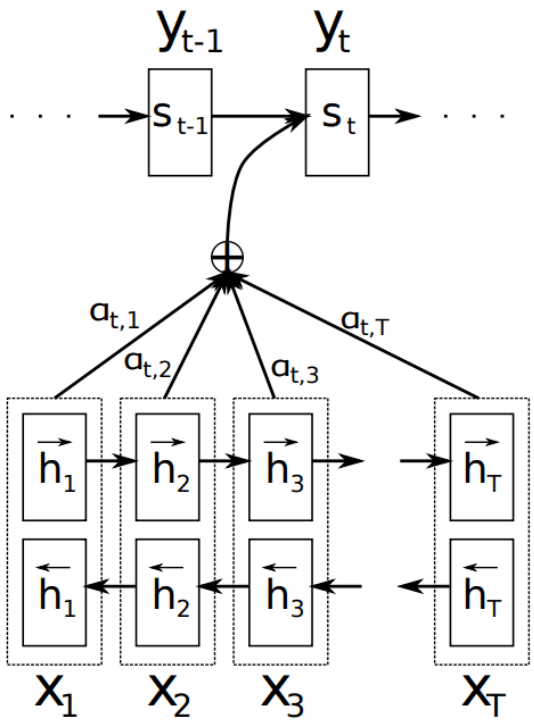


**Figure 2.2**: *Bahdanau Attention Mechanism (Bahdanau et al., 2014)*[22]

where $s_i$ is an recurrent neural network hidden state for time $i$, computed as

$$s_i = f(s_{i-1}, y_{i-1}, c_i) \tag{2.8}$$

Context vector, $c_i$, is computed as

$$c_i = \sum_{j=1}^{T_x} \alpha_{ij} h_j \tag{2.9}$$

The weight $\alpha_{ij}$ of each annotation $h_j$ is computed as follows:

$$\alpha_{ij} = \frac{exp(e_{ij})}{\sum_{k=1}^{T_x} exp(e_i k)} \tag{2.10}$$

where

$$e_{ij} = a(s_{i-1}, h_j) \tag{2.11}$$

### 2.3.2 Transformer-based Models

Vaswani et al. (2017)[58] proposed the Transformer model, an architecture that relies entirely on an attention mechanism to draw global dependencies between input and output. The Transformer model consists of an encoder-decoder architecture and is designed for sequence-to-sequence tasks. The architecture of this model is shown in Figure 2.3. The transformer uses stacked self-attention and point-wise, fully connected layers for both the encoder and decoder. The idea behind this model is to completely handle long-range dependencies between input and output with attention and recurrence. The left side of the image (see 2.3) shows the encoder, which has one layer of multi-head attention followed by a feed forward layer. The right side shows the decoder, which is similar to encoder, but with an additional masked multi-head attention layer. The encoder and decoder blocks are actually multiple identical encoders and decoders with the same number of units stacked on top of each other.

The model uses self-attention to focus on the relevant parts of the input sequence. It performs multiple independent computations in parallel and therefore is referred to as multi-head attention. The multiple outputs are then concatenated and linearly transformed.

#### Encoder for Transformer

The encoder in a transformer[58] consists of a stack of six identical layers where each layer has two sub-layers: a multi-head self-attention mechanism and a position wise, fully connected, feed-forward network. The output of each sub-layer is given by the following equation:

$$output_{subLayer} = LayerNorm(x + Sublayer(x)) \tag{2.12}$$

where $Sublayer(x)$ is the function implemented by the sub-layer.

#### Decoder for Transformer

The decoder, just like the encoder, consists of a stack of six identical layers. However, unlike the encoder, it has three sub-layers, where the additional layer performs multi-head attention

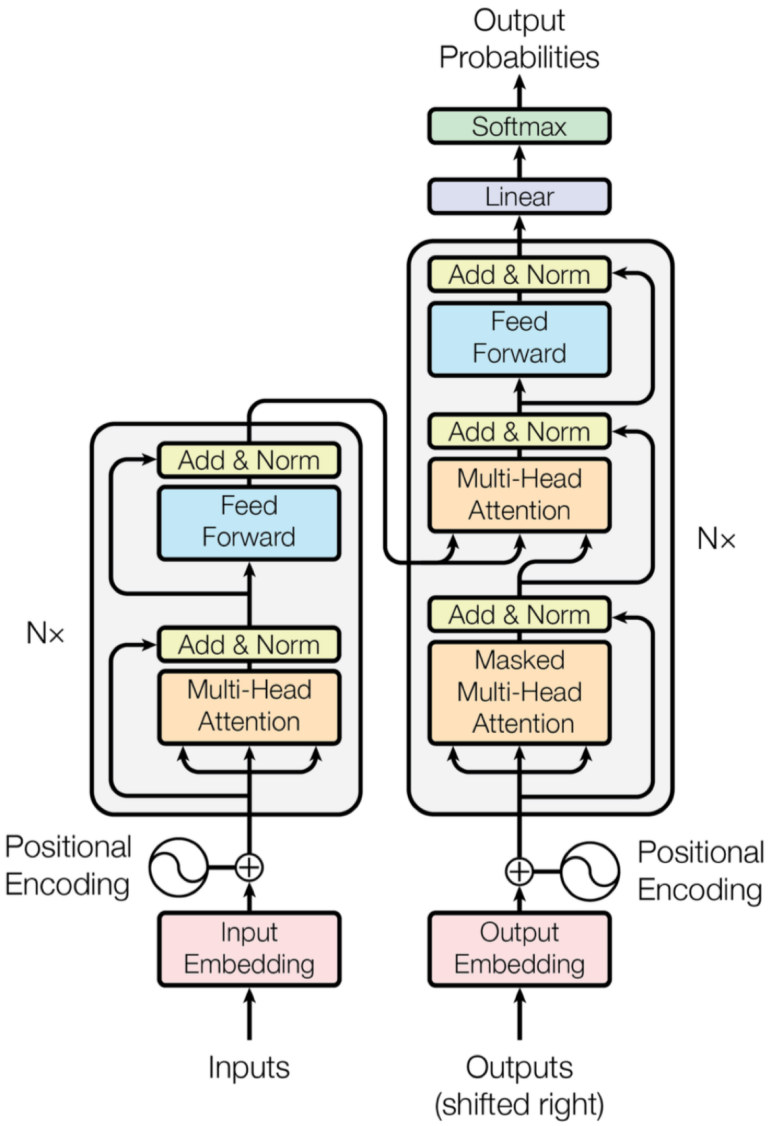


**Figure 2.3**: *Architecture of the Transformer model (Vaswani et al., 2017)*[58]

over the output of the encoder stack. The self-attention sub-layer in the decoder is modified from the encoder version to prevent positions from attending to succeeding positions.

**Scaled Dot-Product Attention:** The input consists of three things: queries, keys with dimension $d_k$, and values with dimension $d_v$. The attention is computed for a set of queries using Equation 2.13:

$$Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d_k}})V \tag{2.13}$$

where Q is a matrix consisting of a set of queries, K is a matrix of keys, and V is a matrix of values.

**Multi-Head Attention:** Transformers use multi-head attention that allows them to jointly attend to information at different positions. Multi-head attention is computed using Equation 2.14:

$$MultiHead(Q, K, V) = Concat(head_1, ..., head_h)W^O \tag{2.14}$$

where $head_i = Attention(QW_i^Q, KW_i^K, VW_i^V)$

### 2.3.3 Text-to-Text Transfer Transformer (T5)

The T5 model was proposed by Raffel et al. (2020)[65] with an architecture closely following the Transformer form proposed by Vaswani et al. (2017)[58]. The T5 model is pre-trained on a data set called colossal clean crawled corpus (C4) introduced by Raffel et al. (2020)[65]. T5 expects the natural language processing task to be re-framed into a unified text-to-text-format where the input and output are always text strings. This text-to-text framework makes the model flexible enough to be used for many natural language processing tasks including machine translation, question answering, summarizing, and natural language processing classification tasks such as sentiment analysis. The architecture of T5 is similar to that of the original Transformer model by Vaswani et al. (2017)[58] with a few exceptions: LayerNorm of Vaswani et al. (2017)[58] is simplified, dropout is added to the feed-forward network, and encoding is changed to relative position encoding instead of sine-based encoding.

In this research, I have fine-tuned the T5 model for question generation. To date, this model has not been used for question generation in the privacy policy domain.

### 2.3.4 Other Transformer Models

There are some other transformer models, which are briefly summarized here. Some of these models will be used in future work for question generation in privacy policy domain.

Devlin et al. (2019)[63] presented a transformer approach called BERT, a self-supervised approach for pre-training a transformer encoder, which can then be fine tuned for any natural language processing task. Chan and Fan (2019)[62] used BERT for question generation, demonstrating that using BERT out of the box does not yield remarkable results for question generation. They proposed two models that restructure BERT and show significant improvements in results over sequence-to-sequence baselines. Tay et al. (2020)[73] provided a comprehensive overview of existing transformer models used across multiple domains.

BERT uses masking for inputs, which causes the model to neglect dependencies between the masked words. Yang et al. (2019)[74] presented XLNet as a solution to the problems with BERT in text classification. Liu et al. (2019b)[75] studied BERT and evaluated the effects of training data size and hyper-parameter tuning. Their study found that BERT is considerably under-trained, proposed some improvements to BERT, and renamed it RoBERTa. RoBERTa modifications include: (1) longer model training with larger data set; (2) removing next sentence prediction; (3) increasing sequence size; and (4) changing masking patterns dynamically.

Lewis et al. (2019)[76] proposed BART, which architecturally is a generalization of many other transformers. BART is a denoising autoencoder for pre-training sequence-to-sequence models. Joshi et al. (2019)[77] proposed another transformer model, named SpanBERT, which extends BERT by: masking contiguous random spans of text, and using the span boundary representations for training. Their work demonstrated how a good pre-training task could have a remarkable impact on the performance of the model.

## 2.4 Corpora for Question Generation

Question generation is regarded as a sub-task of question answering, so any question answering data set can be used to generate questions. Stanford Question Answering Data set or SQuAD (Rajpurkar et al., 2016)[61] is the most popular data set for neural question generation and has been very widely used in existing literature. It consists of over 100,000 factoid questions based on Wikipedia articles with answers as a span of text. A few other data sets

that have been used in literature include: MS MARCO (Nguyen et al., 2016)[78], NewsQA (Trischler et al., 2017)[79], and LearningQ (Chen et al., 2018)[80].

## 2.5 Summary

This chapter presented a survey of question generation literature, along with state-of-the-art deep learning models used for the task. This chapter also presented the classification of question generation methods into two broad categories: rule-based and neural question generation. Neural question generation methods include two kinds of models: sequence-to-sequence and transformer-based. This work uses sequence-to-sequence and transformer-based T5 model for neural question generation in the privacy policy domain. Literature review in neural question generation suggests that to get better performance, auxiliary information needs to be provided to the system. In my research, I focus on providing domain knowledge to the models during training by expressing the knowledge as constraints. The next chapter discusses existing methods in literature for incorporating domain knowledge as part of machine learning models.

# Chapter 3

# Methodology

## 3.1 Domain Knowledge in Machine Learning Models

Linear machine learning models have used constrained conditional models to inject domain specific knowledge as part of training or evaluation. However, in the past few years, some research has included domain knowledge as part of deep learning models. This chapter presents constrained conditional models, relevant research and its application to question answering systems, followed by a discussion of ways to incorporate domain knowledge into deep learning models. This chapter further explores ways to express different kinds of constraints[81].

## 3.2 Constrained Conditional Models

Natural language processing problems can be formulated as integer linear programming, also known as constrained conditional models[82], a learning and inference framework that uses declarative constraints to augment the learning of conditional models. This framework allows us to frame problems as constrained optimization problems, where the objective function comprises learned models subject to problem or domain specific constraints. Injecting prior knowledge about the domain into the learning process in the form of constraints helps models make coherent decisions. Constraints can be written as first order logic expressions.

Formally, a constrained conditional model can be represented using two weight vectors: $\lambda$ and $\rho$. The objective function is defined below:

$$\underset{y \in Y_{valid}}{\operatorname{argmax}} \lambda \cdot F(x, y) - \sum_{i=1}^{K} \rho_i d_{C_i(x)}(x, y) \tag{3.1}$$

where, $\lambda$ is the weight vector for the learning model, and $F(x, y)$ is a collection of classifiers. The term to the left of the minus operator is a standard linear model, and the term to the right models prior knowledge in the form of constraints. $\rho_i$ is the penalty term for violating the constraints, and $d$ is the function that measures the degree to which the constraint $C_i$ is violated in a pair $(x, y)$. The degree of violation can be measured in several ways. Note that Equation 3.1 allows both "hard constraints" and "soft constraints". This objective function can be solved using integer linear programming. Integer linear programming is intractable in limit but has been successfully used; constraints are, after all, sparse. Beam search has also been used as an alternative to find an approximate solution.

### 3.2.1 Existing Work on Constrained Conditional Models

Chang et al. (2008, 2012)[82;83] formalized and generalized the approach introduced by Roth and Yih (2004, 2007)[84;85] to develop a framework known as constrained conditional models. These models allow declarative constraints to be directly injected into the model while maintaining its simplicity and ease of understanding. The constraints are prior knowledge injected into the model to make it more expressive. The authors used constrained conditional models for sequence labeling task in a supervised and semi-supervised setting.

Kundu et al. (2011)[86] surveyed constrained conditional models and used them in an information fusion system. Their work also outlined the advantages of using constrained conditional models: (1) they allow encoding of prior knowledge; (2) constraints are more expressive than features; and (3) the models provide a framework to combine simple models with a small set of constraints to boost model performance.

Ning et al. (2018)[87] proposed a joint framework called Temporal and Causal Reasoning

that uses constrained conditional models and integer linear programming to extract temporal and causal relations between events.

### 3.2.2 Applying Constrained Models to Question Answering

Contractor et al. (2021)[88] labelled the questions from a tourism forum to identify the informative parts of a question. Their work used constrained conditional models with neural networks to form a query that can be used to retrieve the answer from a knowledge base. Aghaebrahimian and Jurčíček (2016)[89] used constrained conditional models for simple open-domain question answering. Their work was based on the research of Bordes et al. (2015)[90] who had stated that the answer to a simple question can be obtained only by knowing one entity and one property. Their work used a knowledge graph; the metadata from the knowledge graph was used as part of their proposed framework, which used a constrained conditional model.

## 3.3 Domain Knowledge in Deep Neural Networks

Incorporating domain knowledge into deep neural models is a way to emulate human reasoning and understanding. This knowledge can be expressed in terms of constraints that can be designed by experts and then integrated into a deep learning model to make it generalize better. Dash et al. (2021)[81] categorized the representation of domain knowledge for deep neural networks as (1) logical constraints and (2) numerical constraints. A recent survey for injecting domain knowledge in neural networks has been presented by Borghesi et al. (2020)[91]. According to their survey, domain knowledge can be represented by algebraic equations: (1) linear and non-linear equations; (2) equality and inequality constraints; (3) logic formulas. Additionally, domain knowledge can also be expressed in the form of graphs, such as knowledge graphs. Borghesi et al. (2020)[91] also provided a classification of ways in which the constraint-expressed domain knowledge can be integrated in the deep learning network: (1) feature space; (2) hypothesis space; (3) data augmentation; (4) regularization

schemes; (5) constraint learning.

### 3.3.1 Logical constraints

Logical constraints can be expressed using: (1) propositional logic and (2) first-order logic. Domain knowledge can be encoded as rules in propositional logic to constrain the deep neural network.

#### Propositional Logic

A proposition in propositional logic is defined as a declarative sentence that is either true or false, but not both. Propositional logic can be used to represent non-Boolean features with Boolean-valued propositions. Domain experts generate domain-specific features expressed in propositional logic to constrain the parameters or structure of a neural model. Using propositional logic to specify constraints to a neural network is not new; it has been around since the early 90s.

Towell et al. (1990)[92] presented the oldest research on this topic, proposing a knowledge-based artificial neural network that included domain knowledge in neural networks. Their research used propositional rules to determine a fixed topological structure for an artificial neural network. Fu (1993)[93] presented similar research, proposing a knowledge-based neural network that used semantic constraints. These approaches were popular at the time, but these neural networks could not learn new rules. Xu et al. (2018)[94] presented more recent research using propositional rules as well as a semantic loss function that measures the difference between neural network output and constraints represented as propositional rules.

#### First-order logic

First-order logic provides more flexibility and representational power than propositional logic. First-order logic makes use of quantifiers and/or relations for generating expressions. Li and Srikumar (2019)[95] introduced a new framework that integrates first-order logic rules into a neural network by converting the rules into differentiable components of the neural net-

work without altering the network architecture completely and without the burden of extra learnable parameters. Their research also revealed three major difficulties encountered while trying to infuse rules into neural networks. These difficulties include mapping predicates in rules to actual nodes in the network. They also found that the network requires encoding that can be differentiated, even though logic can not be differentiated. Moreover, logical rules may introduce cyclic dependencies between the nodes of the computational graph.

Yao et al. (2021)[96] proposed an approach to refine a BERT-based language model with human-provided compositional explanations represented as logic rules. Recently, Silvestri et al. (2021)[97] have studied the effects of adding domain and empirical knowledge to a deep learning network using a combination of semantics-based regularization[98] and constraint programming[99]. Their work has showed that adding domain knowledge at the time of training can significantly improve the output of a deep learning model. Semantic-based regularization builds a multi-layer architecture consisting of kernel machines at the input layer, with the output of the input layer fed to higher layers, thus implementing constraints formed from a fuzzy translation of first-order logic formulas.

Sikka et al. (2020)[100] proposed a framework for automating the generation of deep neural networks. Their framework incorporates user-provided domain knowledge for training the model. They used first-order logic to represent declarative knowledge. Giunchiglia and Lukasiewicz (2021)[101] imposed hard constraints expressed as normal logic rules for the task of hierarchical multi-label classification. Their work proposed a constraint layer on top of a multi-class classification neural network to convey the predictions made on classes lower in the hierarchy to upper classes. Additionally, they proposed a novel loss function to impose the constraints.

### 3.3.2 Numerical Constraints

Numerical constraints can be expressed using (1) loss function; (2) constraints on weight; and (3) regularisation.

### Loss Function

Loss Function has been used in existing work to impose constraints on a neural network by augmenting the loss function with additional penalty terms. Aghaebrahimian (2017)[102], for instance, proposed a constrained deep neural network for a question answering system to select sentences containing the answer. The model selects a subset of sentences that can contain the potential answer to a question, thereby reducing search space. The research accomplished this by defining a loss function that enforces the number of shared patterns between questions and sentences as a hard constraint.

### Weight-based constraints

Weight-based constraints have also been used to add constraint-based knowledge to neural networks. Hu et al. (2016)[103] developed an iterative distillation method that transfers the structured information of logic rules into the weights of neural networks. Another way to incorporate numerical constraints in deep neural networks is to add weight constraints by encoding the domain knowledge as a 'prior' term in the Bayes equation of the Bayesian framework. The priors specify the expectations from the neural networks before they receive any input data. These priors can correspond to penalty terms added to the objective function or any regularization term. Jiang et al. (2020)[104] proposed a different flavor of recurrent neural network that they dubbed as FA-RNN, which combines regular expressions with neural networks. This network is constructed using weighted finite state automata.

### Regularization

Regularization is a another way to add numerical constraints to a neural network. It is a technique that limits the model capacity by adding a penalty term in the objective function, like inclusion of L1 and L2 norms. The consequence of including these norms in the objective function is a less complex model[105].

## 3.4 Summary

This chapter presented a survey of existing methodology that uses domain knowledge and/or empirical knowledge in machine learning models during training or evaluation. First, this chapter provided an overview of constrained conditional models and their application to question answering systems. Then, it provided a survey of approaches that have been used for incorporating knowledge in deep neural networks. This chapter also discussed related work and background for the work presented in Chapter 5 of this dissertation. My research expresses domain and empirical knowledge as logical constraints using first-order logic. Additionally, my work also uses loss function to impose the constraints in deep learning models.

# Chapter 4

# Neural Question Generation using Transformers and Sequence-to-Sequence Models

## 4.1 Introduction

The research described in this chapter [1] focuses on using existing sequence-to-sequence models and a transformer-based approach to generate questions in the privacy policy domain. To date, no effort has been made in this domain to generate questions. This chapter discusses the creation of custom named entity labels for the domain and their use as auxiliary information for sequence-to-sequence and transformer models. It also presents the results, which indicate that adding labels as input improves the performance of the aforementioned models.

[1] A significant portion of this chapter has been accepted in IEEE International Conference on Electrical, Computer and Communication Technologies (ICECCT) 2021.

### 4.1.1 Motivation and Challenges

The goal of the research in this chapter is to enhance existing sequence-to-sequence and transformer-based T5 models to develop an automatic question generation system, which can serve as a sub-task in the development of a full-fledged question answering system in the privacy policy domain. As previously mentioned, such a system can be used to generate frequently asked questions and responses, thereby enabling users to read and grasp privacy policies instead of ignoring them entirely because of their length and perceived complexity. Developing such a system has its challenges, one of which is obtaining a large, well annotated data set because annotating policy documents requires domain experts.

To date, only two data sets have been created in this domain that can be used for question answering (QA) and question generation (QG) tasks. Ahmad et al. (2020)[1] and Ravichander et al. (2019)[106] have published data sets created from privacy policy statements. Table 4.1 presents the details of the two data sets. This work uses the PolicyQA data set from Ahmad et al. (2020)[1], which is comparatively larger than the one by Ravichander et al. (2019)[106]; further, Ahmad et al.'s (2020)[1] data set has been annotated by domain experts.

**Table 4.1**: *Question-Answer Data Sets in Privacy Policy Domain*

| Author | Data Set | No. of Policies | No. of Examples |
|---|---|---|---|
| Ahmad et al. (2020)[1] | PolicyQA | 115 websites | 25017 |
| Ravichander et al. (2019)[106] | PrivacyQA | 35 mobile applications | 3500 |

### 4.1.2 Privacy Policies

Wikipedia defines a privacy policy as "a statement or legal document that states how a company or website collects, handles and processes its customer or visitor data." The document is a well structured legally binding agreement between the user and the service provider that explicitly states the data gathering, data handling, and data processing practices of an orga-

nization. This means that a privacy policy document indicates if and how user data will be shared with others, and whether or not it will be sold to other third party service providers. Hence, to protect their privacy, it is imperative that end-users read and understand these documents prior to agreeing to the terms of service. However, research shows that users generally avoid reading these documents because of the seemingly excessive length of the documents[9 11 13], the perceived complexity of the used language[10 13], and an unwillingness to spend time reading the policies in detail[9].

Some savvy users try to get some idea of the policies by checking the frequently asked questions (FAQ) section. Unfortunately, not all services offer an FAQ section; and even if they do, the FAQ section may not contain many of the expected clarifying question-answer sets. As a result, most users accept the policies without realizing that they are giving away the rights to their personal data, which could include sensitive personal information such as name, age, gender, address, phone number, email address, nationality, religion, race, credit card details, social security or similar national identity number, among other things. With data breaches becoming increasingly common, some users end up suffering when a breach occurs at the organization holding their data. Only after being affected by a breach do they realize the kind of personal information that was being gathered by the service provider.

### 4.1.3 Limitations of Existing Work

Du et al. (2017)[25] were the first to automatically generate questions using sequence-to-sequence models. Prior to their work, sequence-to-sequence models were used for language translation[21]. Since their work in 2017, most research in the field of question generation has used sequence-to-sequence models with modifications to the architecture and inputs. In the past couple of years, some work has been done on transformer-based question generation[59;60].

The performance of the existing basic sequence-to-sequence and transformer-based models on the benchmark SQuAD[61] data set, which is a general purpose data set consisting of Wikipedia articles, is quite different from their performance on PolicyQA,[1], the data set used in this work. The comparison is presented later in this chapter in Table 4.5. The results

indicate that the existing models do not perform well in the privacy policy domain.

This degradation in performance of deep learning models in the privacy policy domain may be due to their unsuitability to deal with a privacy domain specific data set like PolicyQA. It might also be due to the size of the PolicyQA data set. PolicyQA consists of just 25,017 examples, while SQuAD consists of over a 100,000 examples. It could be due to a combination of the two or even other unexplored factors. Whatever may be the case, the existing models are inefficient, in their current state, to work in the privacy policy domain and this research tries to address this issue. This research focuses on automatically generating questions in the privacy policy domain with minimum human involvement using sequence-to-sequence and transformer-based approaches.

### 4.1.4 Objectives and Significance

This chapter asserts that augmenting the input contextual information with custom named entities defined for privacy policy statements gives promising results that lay the foundation for this research. The results suggest that adding further domain-specific information to the deep learning models should boost the question generation performance of the models, as measured by evaluation metrics covered in section 4.5.4. The work presented in this chapter will act as a baseline for the work presented in Chapter 5 of this dissertation.

The novel contributions described in this chapter include (1) generating questions from privacy policy documents in an effort to further encourage development of better question answering systems in the domain; and (2) generating named entities for the privacy policy domain because existing named entity recognition tools fail to label most data in this domain. Augmenting data with these named entities improves the results and can act as a baseline for further research in question generation for privacy policy documents. Existing work in the privacy policy domain has focused on extracting knowledge using either a rule-based system, topic modeling-based systems, or question answering systems. However, question generation has remained completely absent. The contributions of this chapter are summarized below:

- Existing sequence-to-sequence and transformer-based models have not been used for

question generation for privacy policy documents. In fact, question generation in the privacy policy domain has never been done, altogether. This research is the first to do so.

- Existing named entity recognition tools fail to label privacy policy documents due to their complex nature. This research involves the creation of custom labels for named entities in this domain.

- Augmenting input to basic sequence-to-sequence models with these named entity tags helps the models outperform the base models. This is in line with Zhou et al. (2017)[27]. Additionally, the augmented data is also provided as input to the transformer-based model (Raffel et al. (2020)[107]), which improves results over the base model. This is the first time that transformer-based T5 model has been provided with input augmented with named entity tags.

## 4.2 Background and Related Work

### 4.2.1 Related Work

Rule-based systems require deep linguistic knowledge to create hand-crafted rules to generate questions. Mitkov and Ha (2003)[15] used a set of general transformational rules to create multiple-choice tests. Gates (2008)[16] used tree manipulation rules to generate fact-based reading comprehension questions.

Khullar et al. (2018)[17] used a syntax-based system that runs on dependency parse information of the input sentence. Heilman and Smith (2009)[3;18] also included a statistical component to score questions in generating a ranked set of questions about the text. Chali and Hasan (2015)[19] focused on topic-based question generation, using rules based on named entity information and semantic role labels. Recently, Dhole and Manning (2020)[20] used syntactic rules that leverage universal dependencies, lexical information, shallow semantic parsing and custom rules to generate questions.

Rule-based systems are easier to interpret, allow developers greater control over model behavior, and typically require less data to achieve a performance comparable to neural-based models. However, building such systems not only requires deep linguistic knowledge, but is also labor intensive. Moreover, the created rules may be domain specific and thus cannot be generalized.

Deep learning-based approaches, on the other hand, provide an end-to-end solution that is driven by data and not handcrafted rules. Most existing neural question generation systems use a sequence-to-sequence framework (Sutskever et al., 2014[21]) with an attention mechanism (Bahdanau et al., 2014)[22]. They also use context and other auxiliary information as input to the model. Pan et al. (2019)[24] conducted a comprehensive survey of recent advances in question generation.

Du et al.(2017)[25] conducted the earliest research on using neural networks for question generation. Zhou et al. (2017)[27] used the position of the answer along with additional features generated from named entity and part of speech tags. However, these types of answer-aware models have a tendency to include words from answers in the generated questions, resulting in questions of poor quality. Kim et al. (2019) replaced the answer with a special token in the paragraph to prevent answer words from being included in the predicted question.

Another approach in question generation has been to encode answer feature separately using a recurrent neural network (RNN)[28;29]. Song et al. (2018)[29] separately encoded the passage (source text) and answer. Similar to the work of Zhou et al. (2017)[27], Ma et al. (2020)[35] proposed an attention-based sequence-to-sequence model that accepts named entity features (Sang and De Meulder, 2003[36]), part of speech tags (Brill, 1992[37]), case, and answer position features as input. Additionally, they used a pointer-generator network[32] and copy mechanism[38] to better exploit the answer position-aware features. Harrison and Walker (2018)[39] also used linguistic features like named entity recognition, word case, and entity co-reference resolution to build a sequence-to-sequence model with two encoders, one for token level embedding, and the other for sentence level embedding. The research presented in this chapter follows the work of Du et al.(2017)[25] and Zhou et al. (2017)[27] and uses

sequence-to-sequence models for privacy policy domain. However, unlike the work of Zhou et al. (2017)[27], this research does not use the answer or its position, or part of speech tags. This work also creates named entity labels for the privacy policy domain and uses them as auxiliary information for sequence-to-sequence models.

In recent years, some work has been done on question generation using transformer-based approaches. Some of the research in question generation that has used transformers includes: Matsumori et al. (2021)[59] proposed a Unified Questioner Transformer (UniQer) for visual question generation; Scialom et al. (2019)[60] used transformer for answer-agnostic question generation on SQuAD data set; and Chan and Fan (2019)[108] used BERT. This research uses "Text-To-Text Transfer Transformer"(T5) proposed by Raffel et al. (2020)[107] and it also uses named entity tags as auxiliary information for T5.

### 4.2.2 Sequence-to-Sequence Models

Sutskever et al. (2014)[21] proposed sequence-to-sequence models as a domain independent method that could map sequences to sequences. Their work used long short-term memory architecture[70] for converting the input sequence into a fixed-length vector and another LSTM for converting the vector back to a sequence. Sutskever et al. (2014) used this approach for language translation models, but in 2017, Du. et al. (2017)[25] used these models to generate questions. The most common sequence-to-sequence models are encoder-decoder models, which use a bi-directional recurrent neural network[68]. For any given sequence of inputs, represented by $(x_1, x_2, ..., x_N)$, a standard recurrent neural network uses the following equations to compute a sequence of outputs, represented by $(y_1, y_2, ..., y_N)$:

$$h_t = \sigma(W^{hx}x_t + W^{hh}h_{t-1}) \tag{4.1}$$

$$y_t = W^{yh}h_t \tag{4.2}$$

The standard recurrent neural network is unable to handle long term dependencies.

Hochreiter and Schmidhuber (1997)[70] used LSTMs to solve this problem. Sutskever et al. (2014)[21] used an encoder-decoder architecture, where an encoder is a stack of several recurrent units (LSTM units); each stack accepts a single element of an input sequence at each timestamp and propagates forward the information collected from its representation. At each timestamp, a hidden state is calculated using Equation 4.1. Similarly, a decoder is a stack of several recurrent units where each unit accepts the hidden state from the encoder to predict an output at each time stamp using Equation 4.2.

### 4.2.3 Transformer Model: T5

The original transformer[58] comprised an encoder-decoder architecture using self-attention[109]. Self-attention is an attention mechanism that relates different positions of a single sequence to compute a representation of the given sequence, while focusing on the relevant words. T5 is architecturally similar to the original transformer model[58] with a few exceptions: (1) it does not have the Layer Norm bias; (2) the layer normalization is placed outside the residual path; and (3) it uses a different position embedding scheme.

## 4.3 Methodology

This work uses the paragraph as input to the model and does not use answer sequence because some answers provided in the data set are implied by the context. Table 4.2 shows two examples where the answers provided in the data set can be inferred from the context. There are many other examples of answers that do not occur in the passage verbatim. In such a situation, the existing sequence-to-sequence models do not work well because these models use answers or hard code answer positions. As a result, all models in this work are trained in an answer-unaware setting. However, not being answer-aware, makes the models less likely to generate targeted questions.

In this chapter, the performance of basic RNN encoder-decoder models is compared to that of T5 transformer model in the privacy policy domain to see whether domain specific

**Table 4.2**: *Examples from Data Set showing Context-Question-Answer tuple with NER tags in bold*

---

**Website:** TGI Fridays

---

**Context:** The **information** that you provide is collected by **TGI Fridays**. In the case of links to our gift card and guest recognition sites, the **information** you voluntarily provide at those **sites** will only be shared with those service vendors who help **TGI Fridays** administer those **websites** or **mobile application** and the services they provide. In any case, **TGI Fridays** is the lawful "owner" of the **information** and each of these vendors may use the **information** only for the purpose of administering the digital or **mobile application** and its services for **TGI Fridays**, and will take all necessary precautions to protect the **information**. Ownership of any **information** you provide us will be held solely by **TGI Fridays**. We *will not* sell ownership of this **data** to any other company or organization.

**Question:** Does the third party follow the privacy practice?

**Answer:** will not

---

**Website:** Kaleida Health

---

**Context:** **Kaleida Health** may disclose your *health information* to authorized public health officials (or a foreign government agency collaborating with such officials) so they may carry out their public health activities. For example, we may share your health **information** with government officials who are responsible for controlling disease, injury or disability. **Kaleida Health** may also disclose your *health* ***information*** to a person who may have been exposed to a communicable disease or be at risk for contracting or spreading the disease if a law permits us to do so. And finally, **Kaleida Health** may release some health **information** about you to your employer if your employer hires us to provide you with a physical exam and we discover that you have a work-related injury or disease that your employer must know about in order to comply with employment laws.

**Question:** Do my shared data leak my identity?

**Answer:** Health information

---

named entity recognition tags improve the results. The results of these experiments establish the baseline for the research presented in Chapter 5 which focuses on generating constraint-based questions using sequence-to-sequence and transformer-based models.

### 4.3.1 Deep Learning Models

This section describes the models used in this research. First, the encoder and decoder of sequence-to-sequence models are described, followed by attention mechanism, and the T5 model used in this work.

#### Encoder + NER Labels

This work uses Gated Recurrent Unit[71] to build the encoder. This follows the work of Zhou et al. (2017)[27] and concatenates the word embedding vector with the label embedding vector to form the input for the encoder. However, a major distinction between the two is that this work does not use answer position or part of speech tags as lexical features. The encoder is unidirectional, reading input embeddings to produce a hidden vector that serves as input to the decoder. Equation 4.3 is used to compute the output of the update gate, $z_t$, and sigmoid activation function is used to map the output values between 0 and 1. The update gate determines how useful past information is to the current state. Values closer to 1 indicate more past information is incorporated in the network, while values closer to 0 indicate that only recent information is retained.

$$z_t = \sigma(W_z x_t + U_z h_{t-1} + b_z) \tag{4.3}$$

where $W_z$ is the weight matrix applied to input $x_t$, $U_z$ is the weight matrix applied to the hidden vector $h_{t-1}$, and $b_z$ is the bias vector. The next Equation 4.4 shows the output of the reset gate, which allows the model to ignore past information that might be irrelevant in future time-steps.

$$r_t = \sigma(W_r x_t + U_r h_{t-1} + b_r) \tag{4.4}$$

where $W_r$ is the weight matrix for input $x_t$, $u_r$ is the weight matrix for hidden state $h_{t-1}$, and $b_r$ is the bias vector for the reset gate. The next step is computing the output of candidate hidden state, which combines the information from the previous hidden state with the input. The output is given by Equation 4.5.

$$\tilde{h} = tanh(W_h x_i + r_t * U_h h_{t-1} + b_h) \tag{4.5}$$

where $W_h$ is the weight matrix for input $x_i$, $r_t$ is the output of reset gate, $U_h$ is the weight matrix for hidden state $h_{t-1}$, and $b_h$ is the bias vector. The resulting values are mapped between -1 and 1 using the tanh activation function. The encoder GRU hidden state $h_t$ is computed by Equation 4.6.

$$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h} \tag{4.6}$$

### Decoder

This work uses a GRU decoder to decode the context and label information to generate the questions. In question generation, the output sequence is a collection of words that form a question. The GRU decoder uses the previous hidden state $h_{t-1}$ to compute the output $y_t$, as well as its own -idden state $h_t$. Any hidden state $h_t$ is computed using Equation 4.7.

$$h_t = f(W^{hh} h_{t-1}) \tag{4.7}$$

The output $y_t$ at time step $t$ is computed using Equation 4.8:

$$y_t = softmax(W^s h_t) \tag{4.8}$$

### Attention-Based Decoder

This work uses an attention-based[22] GRU decoder to decode the context and label information to generate the questions. Attention mechanism has been discussed in Chapter 2: Section 2.3.1.

**Transformer + NER Labels**

Vaswani et al. (2017)[58] proposed the Transformer model with an architecture that relies entirely on an attention mechanism to draw global dependencies between input and output. This work uses the T5 small model[107] available from the Hugging Face library[110]. The NER labels are concatenated to the context and provided as input to the T5 model. Transformers has been discussed in Chapter 2: Section 2.3.1.

## 4.4 Problem Statement for the Task

In this section the question generation task is formally defined, followed by a brief overview of the neural models used for the task. This section further discusses the custom named entities generated for the privacy policy domain using the lookup method.

### 4.4.1 Problem Statement

The problem of question generation is defined as following:

Given a passage (or context from a policy document), $X_p = (x_1, x_2, ..., x_n)$ as input, the model generates a question, Y = $(y_1, y_2, ..., y_T)$. The goal is to find the best $\bar{Y}$:

$$\bar{Y} = \underset{x}{\text{argmax}} P(Y|X_p) \tag{4.9}$$

where $P(Y|X_p)$ is the conditional log-likelihood of the predicted question sequence y, given the input x. Neither the answer sequence nor its position is used as input to the deep learning models.

### 4.4.2 Named Entities

The proposed model will be a sequence-to-sequence model at its core, with named entities that are custom generated for the privacy policy data set. A description of the named entities

is presented later in Table 4.3. The main idea is that given a sentence of the form: "John, who is an engineer, is travelling to Manhattan" with no answer provided, a human could generate the following questions:

- Who is travelling to Manhattan?
- Where is John travelling?
- Who is John?

As a human, we tend to focus on the person and location in the given example. Standard named entity recognizers like the Stanford NER parser[111] can recognize up to 7 classes: location, person, organization, money, percent, date, and time. Privacy policies are not amenable to using standard parsing tools because they do not include data that can be labeled; they do not include names of people, rarely mention any location, and most organizations listed in the data can not be marked by the Stanford NER parser. This indicates the need to analyze the data to identify key entities as the foci of the privacy document. After the labelling step, applying constraints should force the system to focus on key entities during training. The constraints are set out in the next chapter.

Existing NER tools do not work well for privacy policy documents because they lean towards a more general domain, which renders them useless for labeling legal documents. Existing legal NER tools are application specific: court cases[112][113], or court cases in German[114]. This research define five entity types to extract key information from the privacy policy documents. These labels are presented in Table 4.3 and a more descriptive version is presented in Appendix A.

The lookup method has been used for named entity recognition because of the shortage of training data and human annotators. This method consists of creating a list of names comprising of entities; for example, "data" is marked as CONTENT. Once the list has been created, all mentions of the elements in the list are marked as entities. The lookup method has certain advantages: it is simple to implement, easy to maintain, and does not require any training data. However, it does suffer from false positives, and so it requires special

**Table 4.3**: *Labels for Privacy Policy Domain*

| Label | Description |
|---|---|
| LEGAL | Terms that refer to policies or agreements |
| CONTENT | Information and its collection source like browser, website, cookies, and apps |
| ORG | Names of organizations for whom the policy is written or terms like third party or vendors |
| PERSON | Common nouns like customers, visitors, etc. that represent people to whom the service is being provided |
| URL | Web page for the organization or service provider |

effort to ensure a very comprehensive list.

## 4.5 Experimental Design

### 4.5.1 Data Set Description

Ahmad et al. (2020)[1], created the PolicyQA data set containing 25,017 reading comprehension style examples, from an existing corpus called OPP-115[115]. PolicyQA consists of 115 website privacy policy statements and 714 human-annotated questions. Ahmad et al. (2020) also presented a comparison of their data set with PrivacyQA[106]. One significant difference between the two is the nature of answers: PolicyQA contains a short text span from the policy document as the answer, which reduces the user burden in searching for the required information in a given sentence(s); while PrivacyQA contains answers as a list of sentences, so the user has to interpret the answer based on the retrieved sentences.

Another key difference between the two is that the questions and answers of PolicyQA

were annotated by law students, whereas Amazon Mechanical Turkers annotated PrivacyQA. This seems to contribute to the overall better quality of the PolicyQA data set, and hence it has been used in this research work. PolicyQA is split into training (80%), validation (10%) and test (10%) sets at the paragraph level. Table 4.4 shows the distribution of the data into train/validation/test sets.

**Table 4.4**: *Number of Context-Question-Answer Tuples after random shuffling*

| Item | Number of Items |
|---|---|
| Train Set | 20013 |
| Validation Set | 2502 |
| Test Set | 2502 |

### 4.5.2 Implementation Details

All machine learning models in this research have been implemented using PyTorch version 1.7.1. and trained on Nvidia Tesla v100. The hyper-parameters of the baseline models are tuned on the validation set. Before conducting experiments on the data, the following pre-processing tasks were performed:

- Each word in a sequence was converted to lower case.
- SOS (start of sequence) and EOS (end of sequence) tokens were added to all questions.
- Spelling inconsistencies were fixed. For example, the word "parties" was often misspelled as "parities" in the data set. Errors of this nature were corrected throughout the data set. Also, some shortened words like "info" were expanded to their full length ("information" in the case of "info") to ensure consistency.

The NER tag information was concatenated to the context before it was fed to the GRU encoder as input. The GRU hidden sizes of both the encoder and decoder alternated between 256, 500, 1000, and 2000. During decoding, greedy search and beam search with beam width

of 3 were used. For optimization during training, stochastic gradient descent optimizer[116] was used with a learning rate of 0.001. Teacher forcing[117] was used for training, and the best model was selected using lowest model perplexity on the validation set. All hyper-parameters were also selected on the validation set.

### 4.5.3 Baseline Models

The following baseline question generation models have been selected for comparison with the proposed approach:

- **Seq2Seq**[21] is a vanilla sequence-to-sequence framework that uses encoder-decoder architecture. The context input was not reversed for the experiments.
- **Seq2Seq + attention**[22] is an encoder-decoder architecture with Bahdanau attention mechanism.
- **NQG (2017)**[25] is the first model that uses neural networks for question generation. It uses a global attention-based LSTM encoder-decoder model to map a passage to a question. The best results were achieved using GloVe[118] pre-trained embedding with paragraph level encoder.
- **Transformer-based model (T5)** is a Text-To-Text Transfer Transformer from Raffel et al. (2020)[107]. The T5 model is pre-trained on Colossal Clean Crawled Corpus (or C4 for short)[107]. T5-small model, which consists of 60 million parameters, is fine-tuned for question generation in the privacy-policy domain, given a textual context as input.

### 4.5.4 Evaluation Metrics

All question generation models are evaluated using the following three metrics: Bilingual Evaluation Understudy or BLEU-n (Papineni et al., 2002)[119]; Metric for Evaluation of Translation with Explicit ORdering or METEOR (Lavie & Denkowski, 2009)[120]; and Recall-Oriented Understudy for Gisting Evaluation or ROUGE-L (Lin, 2004)[121]. This work uses

the evaluation package released by Chen et al. (2015)[122] and nlg-eval[123] with the transformer model.

- **BLEU-n** measures the n-gram of the candidate question with n-gram of the reference question to count the number of matches at the corpus level. The value of n ranges from $1 \leq n \leq 4$.
- **METEOR** is based on matching uni-grams between reference and candidate questions taking into account the synonyms, stemming, and paraphrases.
- **ROUGE-L** measures the longest common sub-sequence of words between reference and candidate questions.

A higher score for these metrics denotes better quality in generated questions.

## 4.6 Results and Discussion

The performance comparison of the models on PolicyQA and the benchmark SQuAD are presented in Table 4.5. The results show that NQG[25] performs better on SQuAD than it does on PolicyQA. Similarly, the transformer-based model, T5 performs better in terms of BLEU-4, METEOR and ROUGE-L on SQuAD data set. However, vanilla sequence-to-sequence model does better on PolicyQA in terms of BLEU-4 and METEOR, while giving

**Table 4.5**: *Evaluation Results (in percentage) for SQuAD vs PolicyQA*

| Model | PolicyQA | | | SQuAD | | |
|---|---|---|---|---|---|---|
| | BLEU-4 | METEOR | ROUGE-L | BLEU-4 | METEOR | ROUGE-L |
| Seq2Seq | 5.12 | 14.23 | 30.16 | 4.26 | 9.88 | 29.75 |
| NQG (2017)[25] | 9.94 | 15.54 | 30.63 | **12.28** | **16.62** | **39.75** |
| T5-small[107] | 8.53 | 18.29 | 31.02 | **18.59** | **24.99** | **40.19** |

**Table 4.6**: *Evaluation Results (in percentage) for Baseline Models with Greedy Search*

| Model | BLEU-1 | BLEU-2 | BLEU-3 | BLEU-4 | METEOR | ROUGE-L |
|---|---|---|---|---|---|---|
| Seq2Seq | 27.22 | 15.52 | 8.63 | 5.12 | 14.23 | 30.16 |
| Seq2Seq+attn | 28.09 | 16.00 | 9.86 | 6.30 | 14.96 | **31.84** |
| NQG (2017)[25] | **32.66** | **18.27** | **12.73** | **9.94** | 15.54 | 30.63 |
| T5-small[107] | 31.32 | 17.14 | 11.51 | 8.53 | **18.29** | 31.02 |

**Table 4.7**: *Evaluation Results (in percentage) for Baseline Models with Beam Search*

| Model | BLEU-1 | BLEU-2 | BLEU-3 | BLEU-4 | METEOR | ROUGE-L |
|---|---|---|---|---|---|---|
| Seq2Seq | 25.18 | 13.92 | 7.79 | 3.86 | 16.16 | 30.86 |
| Seq2Seq+attn | **28.29** | **15.97** | **9.72** | 6.36 | 16.15 | 30.02 |
| NQG (2017)[25] | 26.34 | 13.76 | 8.17 | 5.16 | 17.79 | **35.25** |
| T5-small[107] | 28.19 | 14.77 | 9.51 | **6.59** | **18.16** | 28.85 |

a comparable ROUGE-L score. The unsatisfactory performance of existing methods on PolicyQA underpins the motivation behind this research.

Tables 4.6 and 4.7 present a comparison of all baseline models using greedy and beam search, respectively. The performance of sequence-to-sequence and transformer-based (T5) models varies when greedy and beam searches are used for decoding. When greedy search is used for decoding (see 4.6), T5-small model gives the best METEOR score, but the ROUGE-L score is lower than that of attention-based sequence-to-sequence model. When using beam search, METEOR appears to be higher for all sequence-to-sequence models as compared to greedy search. However, BLEU-n and ROUGE-L on average are higher for models when using greedy search for decoding.

Sequence-to-sequence with attention gives slightly better results than vanilla sequence-to-sequence. NQG (2017)[25] performs better than both vanilla sequence-to-sequence and sequence-to-sequence model with attention because it uses GloVe pre-trained embedding.

The results of greedy search decoding show a smooth trend in the METEOR score. ROUGE-L is comparable for vanilla sequence-to-sequence and NQG models. However, the Rouge-L scores for these fall short of T5 and sequence-to-sequence with attention.

As per table 4.7, T5 performs better in terms of BLEU-4 and METEOR as compared to all sequence-to-sequence models. NQG (2017)[25] performs better than T5 and the sequence-to-sequence models, showing a 15.08% increase as compared to its ROUGE-L value for greedy search in Table 4.6. Overall, greedy search produces better METEOR and BLEU-4 scores than beam search.

Table 4.8 presents the results after the context input has been augmented with named entity labels generated for the privacy documents. The results improve over the corresponding baselines: T5-small model with named entity labels shows an increase of 2.46% in METEOR and 3.67% increase in ROUGE-L over the baseline T5 model without custom labels. Vanilla sequence-to-sequence shows a 4.9% improvement in METEOR and 9.6% improvement in ROUGE-L over the baseline. Similarly, sequence-to-sequence models with attention decoder also show a 5.6% improvement in METEOR and a comparable performance in terms of ROUGE-L when named entity labels are added. Additionally, sequence-to-sequence with attention almost shows a 1% point improvement over vanilla sequence-to-sequence for ME-

**Table 4.8**: *Evaluation Results (in percentage) for Models with data augmented with custom labels using Greedy Search*

| Model | BLEU-1 | BLEU-2 | BLEU-3 | BLEU-4 | METEOR | ROUGE-L |
|---|---|---|---|---|---|---|
| Seq2Seq | 27.22 | 15.52 | 8.63 | 5.12 | 14.23 | 30.16 |
| Seq2Seq+attn | 28.09 | 16.00 | 9.86 | 6.30 | 14.96 | 31.84 |
| T5-small[107] | 31.32 | 17.14 | 11.51 | 8.53 | 18.29 | 31.02 |
| Seq2Seq+NER | 24.66 | 14.33 | 8.96 | 5.80 | 14.93 | **33.07** |
| Seq2Seq+attn+NER | 28.68 | 16.24 | 9.95 | 6.87 | 15.81 | 31.51 |
| T5-small + NER | **32.98** | **18.22** | **11.89** | **8.49** | **18.74** | 32.16 |

TEOR with named entity labels. This improvement can also be noticed for BLEU. Overall, the addition of named entity labels as auxiliary information improves the performance of all models.

## 4.7 Summary

This chapter presented experimental results and performance comparison of sequence-to-sequence models and transformer-based T5 model, using both greedy and beam decoding mechanisms. The results recorded in the chapter show that augmenting the input to the deep learning models with custom named entity labels generated for the privacy policy domain provides a boost to the BLEU-n, METEOR, and ROUGE-L scores, as compared to the baselines established in Chapter 4. This indicates that adding additional domain information produces better results, thereby bolstering the motivation behind the work discussed in the next chapter, which covers the use of these entity labels to add constraints to the deep learning models during training. The results presented in this chapter will serve as the baseline for the experiments presented in the following chapter, which is the core of this dissertation.

# Chapter 5

# Constrained Question Generation using Sequence-to-Sequence and Transformer-based models

## 5.1 Introduction

The focus of the work presented in this chapter is on using constraints with existing sequence-to-sequence models and a transformer-based (T5) approach to generate appropriate questions from privacy policy documents. This research is the first to apply constraints to question generation. The motivation behind the use of constraints is to further enhance the results presented in the preceding chapter by creating an approach that could be generalized to any domain.

The success of deep learning models is based on their capability of learning from huge amounts of data. However, deep learning models may produce less than satisfactory results when the size of data set is small or when the nature of data is complex. Since the data set used in this research possesses these qualities, constraints are used to encode domain knowledge in a bid to improve the performance of the deep learning models. Moreover, the use of logical and numerical constraints helps in making the results of the deep learning

models more interpretable. The baselines discussed in the preceding chapter were considered while designing constraints that help encode domain knowledge. Results indicate that adding constraints to deep learning models during training improves the performance, as measured by the evaluation metrics, for all the models used in this study.

The novel contributions of this chapter are outlined below:

- The work discussed in this chapter is the first to use constraint-based learning to generate questions. It analyzes the effects of logical constraints, empirically derived linguistic constraints, and a combination of the two to steer deep learning models towards generating questions that are more relevant to the input text.

- This work has resulted in providing a significant boost to the performance of the models. The results show consistent improvement in the BLEU-n, METEOR, and ROUGE-L scores for all models over the baseline results presented in Chapter 4.

- To the best of my knowledge, this is the first research to perform question generation in the privacy policy domain.

### 5.1.1 Limitations of Existing Work

Deep learning models have produced state-of-the-art results in several tasks across different domains. These models are capable of learning features on their own and tend to perform well with large data sets. However, in the legal space, there are domains like privacy policies where annotated data is scarce. As a result, there is a chance that the size of data set may be inadequate to learn useful patterns, and consequently, the performance of a deep learning models may be sub-optimal. One solution to this problem is to exploit prior domain knowledge and pass it as input to the deep learning model to improve its performance.

In the preceding chapter, the PolicyQA data set was used with existing sequence-to-sequence and T5 models. The performance of the models on PolicyQA was compared to their performance on the benchmark SQuAD data set. The results clearly demonstrated that the performance of the existing models does not translate well to PolicyQA. This lack

of performance could be attributed to either the small size of the PolicyQA data set or its complex nature. The work presented in this chapter focuses on finding ways to include additional knowledge in deep learning models to counteract the effects of a small data set. Exploiting domain knowledge refers to extracting problem specific information from the data that can be provided to the model during the training process to guide it towards learning the correct patterns. This chapter makes use of logical constraints, the formulation of which does not require domain experts. In addition to logical constraints, the baseline results of Chapter 4 were used to design linguistic and empirical constraints to improve the results of the deep learning model.

### 5.1.2 Objectives and Significance

This dissertation proposes to incorporate domain-based knowledge represented as constraints in deep learning models to improve their performance. This work asserts that even without the need of linguistic or domain experts, logical and empirical constraints can be designed to represent knowledge that can force the neural network to generate relevant questions. Existing literature is full of noteworthy work showing that providing additional information to deep learning models can boost performance. The preliminary results from Chapter 4 corroborated this finding. Adding named entity tags alone was able to improve the performance of the deep learning models. The next part of this research focuses on providing a deep learning network with sufficient information in an answer-unaware setting.

This chapter reports the following novel contributions to the state of the field. First, it extends the work in Chapter 4 by adding constraints to the sequence-to-sequence and T5 transformer models. Second, the research creates penalty and reward terms to alter the learning objective for the neural network. Third, the results provide direction for future research, as discussed in Chapter 6. They encourage future research in question generation in the privacy domain as well as constraint-based question generation, in general.

### 5.1.3 Proposed Constraints

The first constraint that was generated for this research is a logical constraint based on human knowledge, expressed using first-order logic. The constraint uses named entity tags, discussed in Chapter 4, to force the deep learning model to include an entity term during training. The idea is to focus on entities given in context to formulate questions. The constraint is expressed as Equation 5.1. It checks if the generated question consists of a named entity term. In the absence of such a term the value of the constraint becomes 1, otherwise it becomes 0. This constraint will be referred to as Constraint 1 in this chapter for brevity.

#### Logical Constraint

Let W be a set of all named entities and $Y$ be a set of all words in a generated question, such that $Y = (y_1, y_2, ..., y_n)$.

The constraint $f(c_1)$ can be defined as follows:

$$f(c_1) = \begin{cases} 0 & \text{if } \exists\ t \in [1, n],\ \text{such that } y_t \in W \\ 1 & \text{if } \forall\ t \in [1, n],\ y_t \notin W \end{cases} \tag{5.1}$$

The next constraint is designed based on empirical results, where models that underperform tend to output the same word, consecutively, more than once. For instance, the generated question could be - "Do you you share share my data?". This constraint checks if two consecutive terms are the same, and if so, it becomes 1, zero otherwise. This constraint will be referred to as Constraint 2 throughout this chapter.

#### Empirical Constraint

Let $Y$ be a set of all words in a generated question, such that $Y = (y_1, y_2, ..., y_n)$.

The constraint $f(c_2)$ can be defined as follows:

$$f(c_2) = \begin{cases} 1 & if\ \exists\ t \in [2, n],\ \text{ such that } y_{t-1} = y_t \\ 0 & if\ \forall\ t \in [2, n],\ y_{t-1} \neq y_t \end{cases} \tag{5.2}$$

**Domain Constraint**

Domain constraints are learned from the context data itself by using the Apriori algorithm[124]to mine frequent-item sets. The used confidence and support values are determined empirically. A key difference between a domain constraint and the aforementioned constraints is that a domain constraint rewards the objective function, instead of imposing a violation penalty.

For any given frequent item set of size $n$, let $Z$ be the set of all words in the item set. A domain constraint $f(c_3)$ can then be defined as follows:

$$f(c_3) = \frac{1}{n}[\lambda_1 x_1 + \lambda_2 x_2 + ... + \lambda_n x_n] \tag{5.3}$$

where $f(c_3) \in [0, 1]$; $\lambda_i$ is an empirical term that assigns weight to the $i^{th}$ term in the item set; and $x_i$ is a binary variable that indicates the presence of the $i^{th}$ word from set Z in a predicted question.

In simple words, this constraint says that if all words occurring in set $Z$ also occur in a question then the objective function should be given maximum reward. However, if no word from set $Z$ appears in the question then no penalty should applied. For this research, only 1 frequent item set was converted to a domain constraint and applied. This constraint will be referred to as Constraint 3 in this chapter.

## 5.2 Related Work

Borghesi et al. (2020)[91] were the first to survey approaches that integrate constraint-based domain knowledge into deep neural networks. Chapter 3 of this dissertation provided an overview of the literature on the inclusion of domain knowledge in deep learning models across different domains and for a variety of deep learning models. This section briefly presents the taxonomy provided by Rueden et al. (2019)[125] to classify knowledge, different forms of representing knowledge, and ways in which it is integrated with the machine learning system.

Rueden et al. (2019)[125] provided three categories for knowledge type: (1) scientific knowledge (formalized and validated via experiments); (2) world knowledge (intuitive and validated by human reasoning); and (3) expert knowledge (knowledge provided by domain experts). This chapter uses scientific and world knowledge to enhance the deep learning models.

Knowledge representation has also been categorized as follows: (1) algebraic equations (as equality or inequality relations); (2) differential equations (describing relationships between functions and their derivatives) ; (3) simulation results (the numerical outcome of a computer simulation); (4) spatial in-variances (properties that do not change under mathematical transformations); (5) logic rules (set of boolean expressions that are combined with logical connectives); (6) knowledge graphs (with vertices usually representing concepts and edges representing relations between nodes); (7) probabilistic relationships (conditional independence or correlation structure); and (8) human feedback (technologies that transform knowledge between users and machines). In this chapter, logical rules represent prior knowledge.

For injecting data into a machine learning system, i.e., for knowledge integration, Rueden et al. (2019)[125] suggest the following ways: (1) inclusion in the form of training data; (2) inclusion as part of deep learning architecture; (3) inclusion as part of learning algorithm via loss function augmented by additional terms to account for the additional domain knowledge; and (4) rejection of the outcomes that do not satisfy the given set of constraints during the

final prediction phase.

The work in this chapter includes knowledge represented as logical rules, as part of the learning algorithm via a loss function augmented by additional penalty terms and a reward term (in case of Constraint 3).

## 5.3 Deep Learning Models

### 5.3.1 Sequence-to-sequence Models with Constraints

This section revisits some of the concepts about encoder and decoder previously discussed in Chapter 4. For further details, refer to Chapter 4: Section 4.3.1.

#### Encoder + NER Labels

This work uses Gated Recurrent Unit[71] to build the encoder. This follows the work of Zhou et al. (2017)[27] and concatenates the word embedding vector with the label embedding vector to form the input for the encoder. However, unlike the model proposed by Zhou et al. (2017)[27], the answer position or part of speech tags are not used as lexical features. The encoder is unidirectional, reading input embeddings to produce a hidden vector that serves as input for the decoder.

#### Decoder

This work uses a GRU decoder to decode the context and label information to generate the questions. In question generation, the output sequence is a collection of words that form a question. The GRU decoder uses the previous hidden state $h_{t-1}$ to compute the output $y_t$, as well as, its own hidden state $h_t$.

#### Attention-Based Decoder

This work uses an attention-based GRU decoder to decode the context and label information to generate the questions.

### 5.3.2 T5 Model with Constraints

Vaswani et al. (2017)[58] proposed the Transformer model, which has an architecture that relies entirely on an attention mechanism to draw global dependencies between input and output. This work uses the T5 small model[107] available from the Hugging Face library[110] and uses the context information to generate questions. The NER labels are added to the context and provided as auxiliary input to the T5 model. Transformers have been discussed in Chapter 2: Section 2.3.1.

## 5.4 Methodology

### 5.4.1 Problem Formulation

The task of question generation is defined as follows:

Given a passage (or context from a policy document), $X_p = (x_1, x_2, ..., x_n)$, as input, the model generates a question, Y = $(y_1, y_2, ..., y_T)$. The goal is to find the best $\bar{Y}$ using Equation 5.4:

$$\bar{Y} = \underset{x}{\text{argmax}} P(Y|X_p) \tag{5.4}$$

where $P(Y|X_p)$ is the conditional log-likelihood of the predicted question sequence y, given the input x. The answer sequence or its position is not used as input to the deep learning models. The goal is to leverage domain knowledge to train a robust model that produces relevant and semantically correct questions. For each constraint that the network has to obey, a loss function, or in some cases a reward function, is defined. The purpose of the loss function is to increase the total loss if the constraint is violated by the network. In the case of a reward function, the purpose is to decrease the loss depending on the degree to which the constraint has been satisfied. A knowledge-based loss term $Loss_C$ can be introduced for each applied constraint into the objective function as follows:

$$loss = \underset{x}{\text{argmin}} \; Loss(Y, \bar{Y}) + \lambda_1 x Loss_{C1}(\bar{Y}) + \lambda_2 y Loss_{C2}(\bar{Y}) - \lambda_3 z Loss_{C3}(\bar{Y}) \tag{5.5}$$

where, x and y are binary variables which are set to 1 when a constraint is not satisfied and 0 otherwise. z denotes the function given by Equation 5.3. The $\lambda$ terms are hyper-parameters that denote the weight of x, y and z in the objective function.

### 5.4.2 Data Preparation

The data was shuffled to create three sets: training, development, and test. The following pre-processing tasks were performed: (1) Each word in a sequence was converted to lowercase; (2) Named entities having two words in their names were hyphenated to reduce the size of the name to one word; (3) SOS (start of sequence) and EOS (end of sequence) tokens were added to all questions; (4) Spelling inconsistencies were fixed. For instance, the word "parties" was often misspelled as "parities" in the data set. Errors of this nature were corrected throughout the data set. Also, some shortened words like "info" were expanded to their full length ("information" in the case of "info") to ensure consistency; and (5) A space was introduced between the last letter in the question and the question mark. For example, "data?" was changed to "data ?".

### 5.4.3 Baselines

The baselines for constraint-based question generation models have already been presented in Chapter 4. They are listed below:

#### Seq2Seq

This is a vanilla sequence-to-sequence framework[21] that uses encoder-decoder architecture. The context input was not reversed for the experiments.

#### Seq2Seq + attention

This is an encoder-decoder architecture with Bahdanau attention mechanism[22].

#### NQG (2017)

NQG[25] uses a global attention-based LSTM encoder-decoder model to map a passage to a question. The best results were achieved using GloVe[118] pre-trained embedding with paragraph level encoder.

#### Transformer-based model (T5)

T5-small model[107] consists of 60 million parameters, and is fine-tuned for question generation in the privacy policy domain, given a textual context with named entities as input.

### 5.4.4 Experimental Setup

All machine learning models in this research have been implemented using PyTorch version 1.7.1. and trained on Nvidia Tesla v100. The hyper-parameters of the baseline models have been tuned on the development set. The GRU hidden sizes of both the encoder and decoder were alternated between 256, 500, 1000, and 2000. During decoding, greedy search was used because results in Chapter 4 showed that greedy search produces better results for this problem. For optimization during training, stochastic gradient descent optimizer was used with a learning rate of 0.001. Teacher forcing[117] was used for training and the best model was selected using lowest model perplexity on the validation set. All hyper-parameters were also selected on the validation set.

### 5.4.5 Evaluation Metrics

All question generation models are evaluated using the following three metrics: Bilingual Evaluation Understudy or BLEU-n (Papineni et al., 2002)[119], Metric for Evaluation of Translation with Explicit ORdering or METEOR (Lavie & Denkowski, 2009)[120], and Recall-

Oriented Understudy for Gisting Evaluation or ROUGE-L (Lin, 2004)[121]. This work uses the evaluation package released by Chen et al. (2015)[122]. **BLEU-n** measures the n-gram of the candidate question with n-gram of the reference question to count the number of matches at the corpus level. The value of n ranges from $1 \leq n \leq 4$. **METEOR** is based on matching uni-grams between reference and candidate questions taking into account the synonyms, stemming, and paraphrases. **ROUGE-L** measures the longest common sub-sequence of words between reference and candidate questions. Ideally, a high score for these metrics denotes better quality in generated questions.

**Table 5.1**: *Evaluation Results (in percentage) for all models with Greedy Search*

| Model | BLEU-1 | BLEU-2 | BLEU-3 | BLEU-4 | METEOR | ROUGE-L |
|---|---|---|---|---|---|---|
| NQG (2017)[25] | 32.66 | 18.27 | 12.73 | 9.94 | 15.54 | 30.63 |
| Seq2Seq | 27.22 | 15.52 | 8.63 | 5.12 | 14.23 | 30.16 |
| Seq2Seq + NER | 24.66 | 14.33 | 8.96 | 5.80 | 14.93 | 33.07 |
| Seq2Seq+NER+C1 | 28.41 | 16.38 | 9.67 | 7.04 | 16.22 | **35.85** |
| Seq2Seq+NER+C2 | 31.08 | **18.67** | **11.68** | **8.53** | 17.79 | 35.18 |
| Seq2Seq+NER+Both | **31.11** | 18.66 | 11.63 | 8.49 | **17.80** | 35.13 |
| Seq2Seq+att | 28.09 | 16.00 | 9.86 | 6.30 | 14.96 | 31.84 |
| Seq2Seq+att+NER | 28.68 | 16.24 | 9.95 | 6.87 | 15.81 | 31.51 |
| Seq2Seq+att+NER+C1 | **31.11** | **18.66** | 11.62 | 8.48 | **17.80** | 35.15 |
| Seq2Seq+att+NER+C2 | 28.40 | 16.38 | 9.67 | 7.04 | 16.22 | **35.85** |
| Seq2Seq+att+NER+All | 31.10 | **18.66** | **11.63** | **8.49** | **17.80** | 35.12 |
| T5-small[107] | 31.32 | 17.14 | 11.51 | 8.53 | 18.29 | 31.02 |
| T5-small+NER[107] | **32.98** | **18.22** | 11.89 | **8.49** | **18.74** | 32.16 |
| T5-small+NER+All[107] | 31.99 | 18.10 | **11.92** | 8.42 | 18.56 | **32.85** |

## 5.5 Results & Discussion

The baseline results are presented in Table 4.8 of Chapter 4. Table 5.1 gives the results from the application of constraints (see 5.1.3) to sequence-to-sequence and T5 models. The first section of the table gives the results of neural question generation (NQG)[25] from Chapter 4. The second section of the table presents the results of vanilla sequence-to-sequence model baselines, followed by sequence-to-sequence with named entity tags, and finally, the effects of constraints on the model.

The results of applying constraint 1 shows improved results across all BLEU-n, METEOR, and ROUGE-L scores. ROUGE-L shows an increase of 8.4% in the results for Constraint 1 over the results for sequence-to-sequence models augmented with named entity tags and an 18.9% increase over the results from the absolute baseline model. The best METEOR score is obtained when both constraints: 1 and 2 are applied to the models. An improvement of 25.1% is observed over the results from the baseline sequence-to-sequence model. A similar improvement is also observed in results across all BLEU-n scores for both constraints: 1 and 2, as well as for a combination of the constraints. The effect of applying either constraint to the model is evident: the model not only outperforms the vanilla sequence-to-sequence model, but also outperforms the NQG (2017)[25] model.

The third section of the table presents the results of sequence-to-sequence with attention mechanism. These results show a similar trend as in the case of vanilla sequence-to-sequence models. All three categories of constraints give improved results, with the the second constraint producing the highest ROUGE-L score and Constraint 1, as well as, a combination of both constraints resulting in the best METEOR score. Similarly, BLEU scores improve over the scores of both, the baseline sequence-to-sequence model with attention, and the model augmented with named entity tags. The impact of constraints is apparent: the vanilla sequence-to-sequence model produced results comparable to the sequence-to-sequence model with attention.

The final section of the table presents the results for transformer-based T5 model. The positive effect of applying a combination of all three constraints is evident on the scores.

**Table 5.2**: *Questions predicted using Transformer Model*

| |
|---|
| **Ground Truth:** How does this website inform users about their policy change?<br><br>**Predicted Question:** Does the website inform users about policy changes? |
| **Ground Truth:** What is the choice scope for users with regards to first-party data collection?<br><br>**Predicted Question:** Do you collect data about users implicitly? |
| **Ground Truth:** Does you collect my information to enhance or personalize my experience?<br><br>**Predicted Question:** Does the company use user's information for customized services? |
| **Ground Truth:** Do you use my information for research?<br><br>**Predicted Question:** Do you collect or use my information? If yes, then what type? |
| **Ground Truth:** Do you share my information with others?<br><br>**Predicted Question:** Does the company share user's information with a third-party? |

However, these scores are lower than the scores produced by sequence-to-sequence models. To further investigate the results of T5 model, a random selection of some of the generated questions is analyzed. These questions are presented in 5.2 along with the actual ground truth questions.

The questions generated by T5 are semantically and syntactically correct and complete. The generated questions are relevant and similar in meaning to the ground truth, for example, the last ground truth question in the table inquires whether the company shares data with others, where "others" refers to any third-party service provider. In fact, the predicted question is more clear and precise in asking whether the company shares data with a third-party. The predicted question in this case is an improvement over the over-simplified version in ground truth. If these two questions were to be evaluated using automatic evaluation metrics used in this study, then BLEU 3 and 4 would come out to be 0.000177% and 5.3077E-7%, respectively. The METEOR is 30% and ROUGE-L is 45.35%. Another predicted question in the table asks whether the company collects user data, and if they do then what type of data is collected. However, the ground truth just asks if user data is used for research. These are some of the examples that suggest that existing evaluation metrics are unable to capture nuances of question generation. In the future, this work would also consider incorporating human evaluation for generated questions.

## 5.6 Conclusion

This chapter presented the core results of this dissertation. The experimental results discussed in this chapter are encouraging and show that constraint-based approach is suitable for future research in question generation. The performance of sequence-to-sequence models was compared with a T5 model using greedy decoding mechanism. The custom named entity labels discussed and used in Chapter 4, were again used in this chapter to add constraints to deep learning models during training. Incorporating the constraints during model training significantly improved the BLEU-n, METEOR, and ROUGE-L scores, thus, paving the way for exploration of more expressive constraints and investigation of different ways of

representing the constraints. The next chapter discusses future directions for this work.

# Chapter 6

# Conclusion and Future Work

## 6.1 Summary of Results

This research used vanilla sequence-to-sequence, sequence-to-sequence with attention mechanism, and a pre-trained T5 model to generate questions. Chapter 4 established the baselines used in this research and also presented the five categories of named entity tags generated for the task of question generation in the privacy policy domain. The BLEU-n, METEOR, and ROUGE-L scores presented in Chapter 4 show a considerable improvement in the performance of both sequence-to-sequence models when they are augmented with named entity tags. The results also indicate an improvement in the transfer-based T5 model. The T5 model produces the best METEOR score, followed by sequence-to-sequence with attention mechanism. The vanilla sequence-to-sequence model earns the lowest METEOR score.

Chan and Fan (2019)[108] had observed that fine-tuning pre-trained language models for text generation and using them to generate questions does not yield satisfactory results. This research corroborates their observations: fine-tuning the pre-trained T5 model for generating questions did not yield satisfactory results. To address this shortcoming, in future research, I will explore the possibility of re-designing the architecture of T5 to better suit question generation.

Chapter 5 discussed the major contributions of this research, describing logical domain-

based constraints and empirical constraints for sequence-to-sequence models. The constraints are implemented by augmenting the loss term to include penalty (and/or reward) terms when the constraints are violated (or satisfied). This creates a modified learning objective for the neural network. Results show an increase in scores over the established baselines of Chapter 4, as well as, an improvement over baseline models augmented with named entity tags. This improvement is seen across all three models used in this work. Thus, these results encourage further exploration of more expressive constraints that enable the neural network to generate diverse, fluent, relevant and semantically correct questions (i.e., improve the overall quality of the questions being generated).

## 6.2 Summary of Contributions

This dissertation asserts the following three novel research contributions:

1. To date, this is the first work on question generation in the privacy policy domain.

2. This research involves generating custom named entity tags for the privacy policy domain. Augmenting the input to the neural models used in this research with these tags improved results over the baseline models. A list of tags is presented in Appendix A of this work.

3. This research presents a constraint-based approach to question generation. It uses a combination of logical and empirically derived constraints to inject knowledge into deep learning models by altering the learning objective. An additional loss term is added as constraint penalty for each constraint used. In some cases, the term may be a reward term that decreases the loss, based on the degree to which a constraint has been satisfied. For all the neural network models used in this work, results show a significant increase in evaluation scores over the baseline results presented in Chapter 4.

4. This work provides an analysis of the effectiveness of the metrics used to evaluate

generated questions.

## 6.3 Future Work

This research opens up several avenues in question generation for future work. The first possibility is to re-design the T5 architecture for question generation. The current T5 model is pre-trained on a general English corpus called English Colossal Clean Crawled Corpus (C4)[65]. However, training T5 from scratch using a data set derived from privacy policy documents may help explain the effects of training and its impact on the performance of the model through a comparison of the newly trained model's results with the results presented in this dissertation (obtained by fine-tuning a pre-trained model).

Exploring other ways of representing domain specific and empirical knowledge as constraints is the next research possibility. Experimenting with additional ways of learning constraints from the data itself also intrigues me. Moreover, since this work does not utilize answer details while generating questions, I would also like to look at answer-aware constraints.

Extending constrained conditional models for the question generation task using deep learning models is yet another possible avenue of research. Since constrained conditional models have already been successfully used in linear models, extending them into sequence-to-sequence models may bring about further improvement.

Finally, a larger data set in the privacy policy domain is needed. To date, only two data sets suitable for the question generation task have been found. The PolicyQA[1] data set used in this work is the larger of the two. However, it has just 25,013 examples. Due to the small size of data sets for most supervised learning tasks in natural language processing, machine translation being an exception, deep learning networks have not been able to improve results in natural language processing tasks by a magnitude similar to that seen in the field of computer vision. Creating a larger data set might help address the issue.

### Short Term Goals

In the near future, I will analyze the ROUGE-L scores from the T5 model in more detail and will attempt to make architectural modifications to the model to further improve the evaluation scores. Additionally, I will compare the performance of T5 with other transformer models for the task of question generation. In recent years, language translation has benefited highly from using pre-trained language models like BERT[63]. Those models have also performed well on tasks such as question answering[63]. However, using them out-of-the-box has not yielded good results in question generation. Pre-training language models on a legal corpus, particularly privacy law, may be worth the effort. Testing the robustness of the approach using a corpus from another domain is another area of immediate focus.

# Bibliography

[1] Wasi Uddin Ahmad, Jianfeng Chi, Yuan Tian, and Kai-Wei Chang. Policyqa: A reading comprehension dataset for privacy policies. *arXiv preprint arXiv:2010.02557*, 2020.

[2] Vasile Rus, Zhiqiang Cai, and Art Graesser. Question generation: Example of a multi-year evaluation campaign. *Proc WS on the QGSTEC*, 2008.

[3] Michael Heilman and Noah A Smith. Good question! statistical ranking for question generation. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 609–617, 2010.

[4] Ghader Kurdi, Jared Leo, Bijan Parsia, Uli Sattler, and Salam Al-Emari. A systematic review of automatic question generation for educational purposes. *International Journal of Artificial Intelligence in Education*, 30(1):121–204, 2020.

[5] Nasrin Mostafazadeh, Ishan Misra, Jacob Devlin, Margaret Mitchell, Xiaodong He, and Lucy Vanderwende. Generating natural questions about an image. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1802–1813, Berlin, Germany, August 2016. Association for Computational Linguistics. doi: 10.18653/v1/P16-1170. URL https://www.aclweb.org/anthology/P16-1170.

[6] Nan Duan, Duyu Tang, Peng Chen, and Ming Zhou. Question generation for question answering. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 866–874, 2017.

[7] Duyu Tang, Nan Duan, Tao Qin, Zhao Yan, and Ming Zhou. Question answering and question generation as dual tasks. *arXiv preprint arXiv:1706.02027*, 2017.

[8] Tong Wang, Xingdi Yuan, and Adam Trischler. A joint model for question answering and question generation. *arXiv preprint arXiv:1706.01450*, 2017.

[9] Jonathan A Obar and Anne Oeldorf-Hirsch. The biggest lie on the internet: Ignoring the privacy policies and terms of service policies of social networking services. *Information, Communication & Society*, 23(1):128–147, 2020.

[10] Fred H Cate. The failure of fair information practice principles. In *Consumer Protection in the Age of the'Information Economy'*, pages 351–388. Routledge, 2016.

[11] Aleecia M McDonald and Lorrie Faith Cranor. The cost of reading privacy policies. *Isjlp*, 4:543, 2008.

[12] Jonathan A Obar and Anne Oeldorf-Hirsch. Clickwrap impact: Quick-join options and ignoring privacy and terms of service policies of social networking services. In *Proceedings of the 8th International Conference on Social Media & Society*, pages 1–5, 2017.

[13] Joel R Reidenberg, Travis Breaux, Lorrie Faith Cranor, Brian French, Amanda Grannis, James T Graves, Fei Liu, Aleecia McDonald, Thomas B Norton, and Rohan Ramanath. Disagreeable privacy policies: Mismatches between meaning and users' understanding. *Berkeley Tech. LJ*, 30:39, 2015.

[14] Joel R Reidenberg, N Cameron Russell, Alexander J Callen, Sophia Qasir, and Thomas B Norton. Privacy harms and the effectiveness of the notice and choice framework. *ISJLP*, 11:485, 2015.

[15] R. Mitkov. Computer-aided generation of multiple-choice tests. In *International Conference on Natural Language Processing and Knowledge Engineering, 2003. Proceedings. 2003*, pages 15–, 2003. doi: 10.1109/NLPKE.2003.1275861.

[16] Donna M Gates. Automatically generating reading comprehension look-back strategy: Questions from expository texts. Technical report, CARNEGIE-MELLON UNIV PITTSBURGH PA SCHOOL OF COMPUTER SCIENCE, 2008.

[17] Payal Khullar, Konigari Rachna, Mukul Hase, and Manish Shrivastava. Automatic question generation using relative pronouns and adverbs. In *Proceedings of ACL 2018, Student Research Workshop*, pages 153–158, Melbourne, Australia, July 2018. Association for Computational Linguistics. doi: 10.18653/v1/P18-3022. URL https://www.aclweb.org/anthology/P18-3022.

[18] Michael Heilman and Noah A Smith. Question generation via overgenerating transformations and ranking. Technical report, Carnegie-Mellon Univ Pittsburgh pa language technologies insT, 2009.

[19] Yllias Chali and Sadid A. Hasan. Towards Topic-to-Question Generation. *Computational Linguistics*, 41(1):1–20, 03 2015. ISSN 0891-2017. doi: 10.1162/COLI_a_00206. URL https://doi.org/10.1162/COLI_a_00206.

[20] Kaustubh Dhole and Christopher D. Manning. Syn-QG: Syntactic and shallow semantic rules for question generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 752–765, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.69. URL https://www.aclweb.org/anthology/2020.acl-main.69.

[21] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. *arXiv preprint arXiv:1409.3215*, 2014.

[22] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

[23] Zhen Cao, Sivanagaraja Tatinati, and Andy WH Khong. Controllable question generation via sequence-to-sequence neural model with auxiliary information. In *2020 International Joint Conference on Neural Networks (IJCNN)*, pages 1–7. IEEE, 2020.

[24] Liangming Pan, Wenqiang Lei, Tat-Seng Chua, and Min-Yen Kan. Recent advances in neural question generation. *arXiv preprint arXiv:1905.08949*, 2019.

[25] Xinya Du, Junru Shao, and Claire Cardie. Learning to ask: Neural question generation for reading comprehension. *arXiv preprint arXiv:1705.00106*, 2017.

[26] Xinya Du and Claire Cardie. Identifying where to focus in reading comprehension for neural question generation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2067–2073, 2017.

[27] Qingyu Zhou, Nan Yang, Furu Wei, Chuanqi Tan, Hangbo Bao, and Ming Zhou. Neural question generation from text: A preliminary study. In *National CCF Conference on Natural Language Processing and Chinese Computing*, pages 662–671. Springer, 2017.

[28] Yanghoon Kim, Hwanhee Lee, Joongbo Shin, and Kyomin Jung. Improving neural question generation using answer separation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 6602–6609, 2019.

[29] Linfeng Song, Zhiguo Wang, Wael Hamza, Yue Zhang, and Daniel Gildea. Leveraging context information for natural question generation. 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 569–574, 2018.

[30] Zhiguo Wang, Wael Hamza, and Radu Florian. Bilateral multi-perspective matching for natural language sentences. *arXiv preprint arXiv:1702.03814*, 2017.

[31] Wenpeng Hu, Bing Liu, Rui Yan, Dongyan Zhao, and Jinwen Ma. Topic-based question generation, 2018. URL https://openreview.net/forum?id=rk3pnae0b.

[32] Caglar Gulcehre, Sungjin Ahn, Ramesh Nallapati, Bowen Zhou, and Yoshua Bengio. Pointing the unknown words. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 140–149, Berlin, Germany, August 2016. Association for Computational Linguistics. doi: 10.18653/v1/P16-1014. URL https://www.aclweb.org/anthology/P16-1014.

[33] Xingwu Sun, Jing Liu, Yajuan Lyu, Wei He, Yanjun Ma, and Shi Wang. Answer-focused and position-aware neural question generation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3930–3939, Brussels, Belgium, October-November 2018. Association for Computational Linguistics. doi: 10.18653/v1/D18-1427. URL https://www.aclweb.org/anthology/D18-1427.

[34] Junmo Kang, Haritz Puerto, and Sung-Hyon Myaeng. Let me know what to ask: Interrogative-word-aware question generation. *ArXiv*, abs/1910.13794, 2019.

[35] Xiyao Ma, Qile Zhu, Yanlin Zhou, and Xiaolin Li. Improving question generation with sentence-level semantic matching and answer position inferring. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8464–8471, 2020.

[36] Erik F Sang and Fien De Meulder. Introduction to the conll-2003 shared task: Language-independent named entity recognition. *arXiv preprint cs/0306050*, 2003.

[37] Eric Brill. A simple rule-based part of speech tagger. Technical report, PENNSYLVANIA UNIV PHILADELPHIA DEPT OF COMPUTER AND INFORMATION SCIENCE, 1992.

[38] Abigail See, Peter J Liu, and Christopher D Manning. 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, 2017.

[39] Vrindavan Harrison and Marilyn Walker. Neural generation of diverse questions using answer focus, contextual and linguistic features. *arXiv preprint arXiv:1809.02637*, 2018.

[40] Xingdi Yuan, Tong Wang, Caglar Gulcehre, Alessandro Sordoni, Philip Bachman, Saizheng Zhang, Sandeep Subramanian, and Adam Trischler. Machine comprehension by text-to-text neural question generation. In *Proceedings of the 2nd Workshop on Representation Learning for NLP*, pages 15–25, Vancouver, Canada, August 2017.

Association for Computational Linguistics. doi: 10.18653/v1/W17-2603. URL https://www.aclweb.org/anthology/W17-2603.

[41] Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3-4):229–256, 1992.

[42] Kaichun Yao, Libo Zhang, Tiejian Luo, Lili Tao, and Yanjun Wu. Teaching machines to ask questions. In *IJCAI*, pages 4546–4552, 2018.

[43] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc., 2014. URL https://proceedings.neurips.cc/paper/2014/file/5ca3e9b122f61f8f06494c97b1afccf3-Paper.pdf.

[44] Yu Chen, Lingfei Wu, and Mohammed J Zaki. Reinforcement learning based graph-to-sequence model for natural question generation. *arXiv preprint arXiv:1908.04942*, 2019.

[45] Yuxi Xie, Liangming Pan, Dongzhe Wang, Min-Yen Kan, and Yansong Feng. Exploring question-specific rewards for generating deep questions. *arXiv preprint arXiv:2011.01102*, 2020.

[46] Liuyin Wang, Zihan Xu, Zibo Lin, Haitao Zheng, and Ying Shen. Answer-driven deep question generation based on reinforcement learning. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 5159–5170, Barcelona, Spain (Online), December 2020. International Committee on Computational Linguistics. doi: 10.18653/v1/2020.coling-main.452. URL https://www.aclweb.org/anthology/2020.coling-main.452.

[47] Liangming Pan, Yuxi Xie, Yansong Feng, Tat-Seng Chua, and Min-Yen Kan. Semantic graphs for generating deep questions. *arXiv preprint arXiv:2004.12704*, 2020.

[48] Hady Elsahar, Christophe Gravier, and Frederique Laforest. Zero-shot question generation from knowledge graphs for unseen predicates and entity types. 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 218–228, New Orleans, Louisiana, June 2018. Association for Computational Linguistics. doi: 10.18653/v1/N18-1020. URL https://www.aclweb.org/anthology/N18-1020.

[49] Hugo Larochelle, Dumitru Erhan, and Yoshua Bengio. Zero-data learning of new tasks. In *AAAI*, volume 1, page 3, 2008.

[50] Ming-Wei Chang, Lev-Arie Ratinov, Dan Roth, and Vivek Srikumar. Importance of semantic representation: Dataless classification. In *Aaai*, volume 2, pages 830–835, 2008.

[51] Sen Hu, Lei Zou, and Zhanxing Zhu. How question generation can help question answering over knowledge base. In *CCF International Conference on Natural Language Processing and Chinese Computing*, pages 80–92. Springer, 2019.

[52] Vishwajeet Kumar, Yuncheng Hua, Ganesh Ramakrishnan, Guilin Qi, Lianli Gao, and Yuan-Fang Li. Difficulty-controllable multi-hop question generation from knowledge graphs. In *International Semantic Web Conference*, pages 382–398. Springer, 2019.

[53] Sheng Bi, Xiya Cheng, Yuan-Fang Li, Yongzhen Wang, and Guilin Qi. Knowledge-enriched, type-constrained and grammar-guided question generation over knowledge bases. *arXiv preprint arXiv:2010.03157*, 2020.

[54] Shijie Zhang, Lizhen Qu, Shaodi You, Zhenglu Yang, and Jiawan Zhang. Automatic generation of grounded visual questions. *arXiv preprint arXiv:1612.06530*, 2016.

[55] Unnat Jain, Ziyu Zhang, and Alexander G Schwing. Creativity: Generating diverse questions using variational autoencoders. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6485–6494, 2017.

[56] Ranjay Krishna, Michael Bernstein, and Li Fei-Fei. Information maximizing visual question generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2008–2018, 2019.

[57] Pushkar Shukla, Carlos E. L. Elmadjian, Richika Sharan, Vivek Kulkarni, M. Turk, and William Yang Wang. What should i ask? using conversationally informative rewards for goal-oriented visual dialog. *ArXiv*, abs/1907.12021, 2019.

[58] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *arXiv preprint arXiv:1706.03762*, 2017.

[59] Shoya Matsumori, Kosuke Shingyouchi, Yuki Abe, Yosuke Fukuchi, Komei Sugiura, and Michita Imai. Unified questioner transformer for descriptive question generation in goal-oriented visual dialogue. *arXiv preprint arXiv:2106.15550*, 2021.

[60] Thomas Scialom, Benjamin Piwowarski, and Jacopo Staiano. Self-attention architectures for answer-agnostic neural question generation. In *Proceedings of the 57th annual meeting of the Association for Computational Linguistics*, pages 6027–6032, 2019.

[61] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas, November 2016. Association for Computational Linguistics. doi: 10.18653/v1/D16-1264. URL https://www.aclweb.org/anthology/D16-1264.

[62] Ying-Hong Chan and Yao-Chung Fan. A recurrent bert-based model for question generation. In *Proceedings of the 2nd Workshop on Machine Reading for Question Answering*, pages 154–162, 2019.

[63] J. Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*, 2019.

[64] Stalin Varanasi Saadullah Amin Günter Neumann. Copybert: A unified approach to question generation with self-attention. *ACL 2020*, page 25, 2020.

[65] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21 (140):1–67, 2020. URL http://jmlr.org/papers/v21/20-074.html.

[66] Yao Zhao, Xiaochuan Ni, Yuanyuan Ding, and Qifa Ke. Paragraph-level neural question generation with maxout pointer and gated self-attention networks. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3901–3910, 2018.

[67] Li Dong, Nan Yang, Wenhui Wang, Furu Wei, Xiaodong Liu, Yu Wang, Jianfeng Gao, Ming Zhou, and Hsiao-Wuen Hon. Unified language model pre-training for natural language understanding and generation. *arXiv preprint arXiv:1905.03197*, 2019.

[68] Mike Schuster and Kuldip K Paliwal. Bidirectional recurrent neural networks. *IEEE transactions on Signal Processing*, 45(11):2673–2681, 1997.

[69] Kyunghyun Cho, Bart Van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. On the properties of neural machine translation: Encoder-decoder approaches. *arXiv preprint arXiv:1409.1259*, 2014.

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

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

[72] Jianshu Zhang, Jun Du, and Lirong Dai. A gru-based encoder-decoder approach with attention for online handwritten mathematical expression recognition. In *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, volume 1, pages 902–907. IEEE, 2017.

[73] Yi Tay, Mostafa Dehghani, Dara Bahri, and Donald Metzler. Efficient transformers: A survey. *arXiv preprint arXiv:2009.06732*, 2020.

[74] Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. Xlnet: Generalized autoregressive pretraining for language understanding. *Advances in neural information processing systems*, 32, 2019.

[75] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, M. Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *ArXiv*, abs/1907.11692, 2019.

[76] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*, 2019.

[77] Mandar Joshi, Danqi Chen, Yinhan Liu, Daniel S Weld, Luke Zettlemoyer, and Omer Levy. Spanbert: Improving pre-training by representing and predicting spans. *Transactions of the Association for Computational Linguistics*, 8:64–77, 2020.

[78] Tri Nguyen, Mir Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, and Li Deng. Ms marco: A human generated machine reading comprehension dataset. November 2016. URL https://www.microsoft.com/en-us/research/publication/ms-marco-human-generated-machine-reading-comprehension-dataset/.

[79] Adam Trischler, Tong Wang, Xingdi Yuan, Justin Harris, Alessandro Sordoni, Philip Bachman, and Kaheer Suleman. NewsQA: A machine comprehension dataset. In

*Proceedings of the 2nd Workshop on Representation Learning for NLP*, pages 191–200, Vancouver, Canada, August 2017. Association for Computational Linguistics. doi: 10.18653/v1/W17-2623. URL https://www.aclweb.org/anthology/W17-2623.

[80] Guanliang Chen, Jie Yang, Claudia Hauff, and Geert-Jan Houben. Learningq: a large-scale dataset for educational question generation. In *Proceedings of the International AAAI Conference on Web and Social Media*, volume 12, 2018.

[81] Tirtharaj Dash, Sharad Chitlangia, Aditya Ahuja, and Ashwin Srinivasan. Incorporating domain knowledge into deep neural networks. *arXiv preprint arXiv:2103.00180*, 2021.

[82] M Chang, Lev Ratinov, and Dan Roth. Constraints as prior knowledge. In *ICML workshop on prior knowledge for text and language processing*, pages 32–39. ignore={Association for Computational Linguistics}, 2008.

[83] Ming-Wei Chang, Lev Ratinov, and Dan Roth. Structured learning with constrained conditional models. *Machine learning*, 88(3):399–431, 2012.

[84] Dan Roth and Wen-tau Yih. A linear programming formulation for global inference in natural language tasks. In *Proceedings of the Eighth Conference on Computational Natural Language Learning (CoNLL-2004) at HLT-NAACL 2004*, pages 1–8, Boston, Massachusetts, USA, May 6 - May 7 2004. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/W04-2401.

[85] Dan Roth and Wen-tau Yih. Global inference for entity and relation identification via a linear programming formulation. *Introduction to statistical relational learning*, pages 553–580, 2007.

[86] Gourab Kundu, Dan Roth, and Rajhans Samdani. Constrained conditional models for information fusion. In *14th International Conference on Information Fusion*, pages 1–8. IEEE, 2011.

[87] Qiang Ning, Zhili Feng, Hao Wu, and Dan Roth. Joint Reasoning for Temporal and Causal Relations. In *Proc. of the Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 2278–2288, Melbourne, Australia, 7 2018. Association for Computational Linguistics. URL http://cogcomp.org/papers/NingFeWuRo18.pdf.

[88] Danish Contractor, Barun Patra, Parag Singla, et al. Constrained bert bilstm crf for understanding multi-sentence entity-seeking questions. *Natural Language Engineering*, 27(1):65–87, 2021.

[89] Ahmad Aghaebrahimian and Filip Jurčíček. Constraint-based open-domain question answering using knowledge graph search. In Petr Sojka, Aleš Horák, Ivan Kopeček, and Karel Pala, editors, *Text, Speech, and Dialogue*, pages 28–36, Cham, 2016. Springer International Publishing. ISBN 978-3-319-45510-5.

[90] Antoine Bordes, Nicolas Usunier, Sumit Chopra, and Jason Weston. Large-scale simple question answering with memory networks. *arXiv preprint arXiv:1506.02075*, 2015.

[91] Andrea Borghesi, Federico Baldo, and Michela Milano. Improving deep learning models via constraint-based domain knowledge: a brief survey. *arXiv preprint arXiv:2005.10691*, 2020.

[92] Geoffrey G Towell and Jude W Shavlik. Knowledge-based artificial neural networks. *Artificial intelligence*, 70(1-2):119–165, 1994.

[93] Li-Min Fu. Knowledge-based connectionism for revising domain theories. *IEEE Transactions on Systems, Man, and Cybernetics*, 23(1):173–182, 1993.

[94] Jingyi Xu, Zilu Zhang, Tal Friedman, Yitao Liang, and Guy Broeck. A semantic loss function for deep learning with symbolic knowledge. In *International conference on machine learning*, pages 5502–5511. PMLR, 2018.

[95] Tao Li and Vivek Srikumar. Augmenting neural networks with first-order logic. *arXiv preprint arXiv:1906.06298*, 2019.

[96] Huihan Yao, Ying Chen, Qinyuan Ye, Xisen Jin, and Xiang Ren. Refining neural networks with compositional explanations. *arXiv preprint arXiv:2103.10415*, 2021.

[97] Mattia Silvestri, Michele Lombardi, and Michela Milano. Injecting domain knowledge in neural networks: a controlled experiment on a constrained problem. In *International Conference on Integration of Constraint Programming, Artificial Intelligence, and Operations Research*, pages 266–282. Springer, 2021.

[98] Michelangelo Diligenti, Marco Gori, and Claudio Sacca. Semantic-based regularization for learning and inference. *Artificial Intelligence*, 244:143–165, 2017.

[99] Francesca Rossi, Peter Van Beek, and Toby Walsh. *Handbook of constraint programming*. Elsevier, 2006.

[100] Karan Sikka, Andrew Silberfarb, John Byrnes, Indranil Sur, Ed Chow, Ajay Divakaran, and Richard Rohwer. Deep adaptive semantic logic (dasl): Compiling declarative knowledge into deep neural networks. *arXiv preprint arXiv:2003.07344*, 2020.

[101] Eleonora Giunchiglia and Thomas Lukasiewicz. Multi-label classification neural networks with hard logical constraints. *arXiv preprint arXiv:2103.13427*, 2021.

[102] Ahmad Aghaebrahimian. Constrained deep answer sentence selection. In *International Conference on Text, Speech, and Dialogue*, pages 57–65. Springer, 2017.

[103] Zhiting Hu, Xuezhe Ma, Zhengzhong Liu, Eduard Hovy, and Eric Xing. Harnessing deep neural networks with logic rules. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2410–2420, Berlin, Germany, August 2016. Association for Computational Linguistics. doi: 10.18653/v1/P16-1228. URL https://aclanthology.org/P16-1228.

[104] Chengyue Jiang, Yinggong Zhao, Shanbo Chu, Libin Shen, and Kewei Tu. Cold-start and interpretability: Turning regular expressions into trainable recurrent neural networks. In *Proceedings of the 2020 Conference on Empirical Methods in Natural*

*Language Processing (EMNLP)*, pages 3193–3207, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.258. URL https://aclanthology.org/2020.emnlp-main.258.

[105] Jan Kukačka, Vladimir Golkov, and Daniel Cremers. Regularization for deep learning: A taxonomy. *arXiv preprint arXiv:1710.10686*, 2017.

[106] Abhilasha Ravichander, Alan W Black, Shomir Wilson, Thomas Norton, and Norman Sadeh. Question answering for privacy policies: Combining computational and legal perspectives. 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 4947–4958, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1500. URL https://www.aclweb.org/anthology/D19-1500.

[107] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21 (140):1–67, 2020. URL http://jmlr.org/papers/v21/20-074.html.

[108] Ying-Hong Chan and Yao-Chung Fan. A recurrent BERT-based model for question generation. In *Proceedings of the 2nd Workshop on Machine Reading for Question Answering*, pages 154–162, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-5821. URL https://aclanthology.org/D19-5821.

[109] Jianpeng Cheng, Li Dong, and Mirella Lapata. Long short-term memory-networks for machine reading. *arXiv preprint arXiv:1601.06733*, 2016.

[110] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu,

Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online, October 2020. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/2020.emnlp-demos.6.

[111] Jenny Rose Finkel, Trond Grenager, and Christopher D Manning. Incorporating non-local information into information extraction systems by gibbs sampling. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 363–370, 2005.

[112] Stavroula Skylaki, Ali Oskooei, Omar Bari, Nadja Herger, and Zac Kriegman. Named entity recognition in the legal domain using a pointer generator network. *arXiv preprint arXiv:2012.09936*, 2020.

[113] Christopher Dozier, Ravikumar Kondadadi, Marc Light, Arun Vachher, Sriharsha Veeramachaneni, and Ramdev Wudali. Named entity recognition and resolution in legal text. In *Semantic Processing of Legal Texts*, pages 27–43. Springer, 2010.

[114] Elena Leitner, Georg Rehm, and Julian Moreno-Schneider. Fine-grained named entity recognition in legal documents. In Maribel Acosta, Philippe Cudré-Mauroux, Maria Maleshkova, Tassilo Pellegrini, Harald Sack, and York Sure-Vetter, editors, *Semantic Systems. The Power of AI and Knowledge Graphs*, pages 272–287, Cham, 2019. Springer International Publishing. ISBN 978-3-030-33220-4.

[115] Shomir Wilson, Florian Schaub, Aswarth Abhilash Dara, Frederick Liu, Sushain Cherivirala, Pedro Giovanni Leon, Mads Schaarup Andersen, Sebastian Zimmeck, Kanthashree Mysore Sathyendra, N. Cameron Russell, Thomas B. Norton, Eduard Hovy, Joel Reidenberg, and Norman Sadeh. The creation and analysis of a website privacy policy corpus. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1330–1340, Berlin, Germany,

August 2016. Association for Computational Linguistics. doi: 10.18653/v1/P16-1126. URL https://www.aclweb.org/anthology/P16-1126.

[116] Sebastian Ruder. An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:1609.04747*, 2016.

[117] Ronald J Williams and David Zipser. A learning algorithm for continually running fully recurrent neural networks. *Neural computation*, 1(2):270–280, 1989.

[118] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, 2014. URL http://www.aclweb.org/anthology/D14-1162.

[119] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318, 2002.

[120] Alon Lavie and Michael J Denkowski. The meteor metric for automatic evaluation of machine translation. *Machine translation*, 23(2-3):105–115, 2009.

[121] Chin-Yew Lin. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81, 2004.

[122] Xinlei Chen, Hao Fang, Tsung-Yi Lin, Ramakrishna Vedantam, Saurabh Gupta, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco captions: Data collection and evaluation server. *arXiv preprint arXiv:1504.00325*, 2015.

[123] Shikhar Sharma, Layla El Asri, Hannes Schulz, and Jeremie Zumer. Relevance of unsupervised metrics in task-oriented dialogue for evaluating natural language generation. *CoRR*, abs/1706.09799, 2017. URL http://arxiv.org/abs/1706.09799.

[124] Rakesh Agrawal, Ramakrishnan Srikant, et al. Fast algorithms for mining association

rules. In *Proc. 20th int. conf. very large data bases, VLDB*, volume 1215, pages 487–499. Citeseer, 1994.

[125] Laura von Rueden, Sebastian Mayer, Katharina Beckh, Bogdan Georgiev, Sven Giesselbach, Raoul Heese, Birgit Kirsch, Julius Pfrommer, Annika Pick, Rajkumar Ramamurthy, et al. Informed machine learning–a taxonomy and survey of integrating knowledge into learning systems. *arXiv preprint arXiv:1903.12394*, 2019.

# Appendix A

# Named Entity Recognition for Privacy Policy Domain

**Table A.1**: *Labels with entities for Privacy Policy Domain*

| Label | Description |
|---|---|
| LEGAL | policy, agreement, privacy-policy, privacy-statement, privacy-control, privacy-setting |
| CONTENT | information, data, website, cookie, social-media, mobile-app, mobile-application, social-network, browser, profile, account, username, password, online-activity, ip-address, device-id, survey, location, health, dnt, service, opt-out |
| ORG | Names of organizations for whom the policy is written or terms like third party or vendors. A comprehensive list is presented on the next page. |
| PERSON | user, subscriber, customer, citizen, person, children, resident, audience, visitor |
| URL | Web page for the organization or service provider |

**Table A.2**: *Labels with entities for Privacy Policy Domain*

| Label | Description |
|---|---|
| ORG | third-party, first-party, company, youtube, tgi-fridays, reddit, motley-fool, fool, rockstar-games, military, sidearmsports, dairyqueen, allstate, google, lynda, tgifridays, enthusiastnetwork, pbs, disinfo, washingtonpost, citizen, foxsports, taylorswift, coffeereview, dccd, barnesandnoble, www, solarviews, voxmedia, thehill, chasepaymentech, abcnews, everydayhealth, sportsreference, bankofamerica, liquor, nytimes, stlouisfed, geocaching, meredith, gamestop, esquire, restaurantnews, wsmv, dailynews, lids, earthkam, foodallergy, miaminewtimes, boardgamegeek, sheknows, tangeroutlet, usa, instagram, vikings, imdb, ifsa-butler, uh, ocregister, latinpost, playstation, minecraft, jibjab, randomhouse, ironhorsevineyards, aol, austincc, wnep, thefreedictionary, sltrib, freep, steampowered, lodgemfg, uptodate, fortune, msn, redorbit, cincymuseum, tulsaworld, fredericknewspost, timeinc, newsbusters, abita, ticketmaster, theatlantic, sci-news, yahoo, style, adweek, cariboucoffee, kaleidahealth, buffalowildwings, post-gazette, internetbrands, mlb, eatchicken, ted, naturalnews, cbsinteractive, washingtonian, dogbreedinfo, walmart, neworleansonline, mohegansun, honda, communitycoffee, nbcuniversal, sciencemag, education, kraftrecipes, rockstargames, acbj, opensecrets, amazon, archives, gawker, reference, si, dailyillini, gwdocs, highgearmedia, zacks |