DISSERTATION

BUILDING NATURAL LANGUAGE GENERATION AND UNDERSTANDING SYSTEMS IN DATA CONSTRAINED SETTINGS

Improving systems when limited or no training examples are available

THESIS APPROVED BY THE DEPARTMENT OF COMPUTER SCIENCE
Technische Universität Kaiserslautern
for the award of the Doctoral Degree

DOCTOR OF ENGINEERING (DR.-ING.)

to

Shailza Jolly

Date of the Defense: 25.11.2022
Dean: Prof. Dr. Jens Schmitt

Reviewers:
Prof. Dr. Andreas Dengel
Prof. Dr. Sebastian Möller

TECHNISCHE UNIVERSITÄT
KAISERSLAUTERN

DE-386

Dedicated to the power that kept me going!

ABSTRACT

In recent years, deep learning has made substantial improvements in various fields like image understanding, Natural Language Processing (NLP), etc. These huge advancements have led to the release of many commercial applications which aim to help users carry out their daily tasks. Personal digital assistants are one such successful application of NLP, having a diverse userbase from all age groups. NLP tasks like Natural Language Understanding (NLU) and Natural Language Generation (NLG) are core components for building these assistants. However, like any other deep learning model, the growth of NLU & NLG models is directly coupled with tremendous amounts of training examples, which are expensive to collect due to annotator costs. Therefore, this work investigates the methodologies to build NLU and NLG systems in a data-constrained setting.

We evaluate the problem of limited training data in multiple scenarios like limited or no data available when building a new system, availability of a few labeled examples when adding a new feature to an existing system, and changes in the distribution of test data during the lifetime of a deployed system.

Motivated by the standard methods to handle data-constrained settings, we propose novel approaches to generate data and exploit latent representations to overcome performance drops emerging from limited training data. We propose a framework to generate high-quality synthetic data when few training examples are available for a newly added feature for dialogue agents. Our *interpretation-to-text* model uses existing training data for bootstrapping new features and improves the accuracy of downstream tasks of intent classification and slot labeling. Following, we study a few-shot setting and observe that generation systems face a *low semantic coverage* problem. Hence, we present an unsupervised NLG algorithm that ensures that all relevant semantic information is present in the generated text.

We also study to see if we really need all training examples for learning a generalized model. We propose a data selection method that selects the most informative training examples to train Visual Question Answering (VQA) models without erosion of accuracy. We leverage the already available inter-annotator agreement and design a diagnostic tool, called (EaSe), that leverages the entropy and semantic similarity of answer patterns.

Finally, we discuss two empirical studies to understand the feature space of VQA models and show how language model pre-training and exploiting multimodal embedding space allows for building data constrained models ensuring minimal or no accuracy losses.

PUBLICATIONS RESULTING FROM THIS THESIS

Parts of the research and material (including figures, tables and algorithms) in this thesis have already been published in (or accepted in):

[1] Stanislav Frolov, Shailza Jolly, Jörn Hees, and Andreas Dengel. "Leveraging visual question answering to improve text-to-image synthesis." In: *Proceedings of the Second Workshop on Beyond Vision and LANguage: inTEgrating Real-world kNowledge (LANTERN)*. 2020, pp. 17–22.

[2] Shailza Jolly, Tobias Falke, Caglar Tirkaz, and Daniil Sorokin. "Data-efficient paraphrase generation to bootstrap intent classification and slot labeling for new features in task-oriented dialog systems." In: *Proceedings of the 28th International Conference on Computational Linguistics: Industry Track*. 2020, pp. 10–20.

[3] Shailza Jolly and Shubham Kapoor. "Can Pre-training help VQA with Lexical Variations?" In: *Findings of the Association for Computational Linguistics: EMNLP 2020*. 2020, pp. 2863–2868.

[4] Shailza Jolly, Sebastian Palacio, Joachim Folz, Federico Raue, Jorn Hees, and Andreas Dengel. "P≈NP, at least in Visual Question Answering." In: *2020 25th International Conference on Pattern Recognition (ICPR)*. IEEE Computer Society. 2021, pp. 2748–2754.

[5] Shailza Jolly, Sandro Pezzelle, and Moin Nabi. "EaSe: A Diagnostic Tool for VQA Based on Answer Diversity." In: *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 2021, pp. 2407–2414.

[6] Shailza Jolly, Zi Xuan Zhang, Andreas Dengel, and Lili Mou. "Search and Learn: Improving Semantic Coverage for Data-to-Text Generation." In: *Proceedings of the 36th AAAI Conference on Artificial Intelligence*. 2022.

ACKNOWLEDGMENTS

I am very grateful to submit this thesis dedicated to completing my doctoral studies. First, I thank my Ph.D. advisor, Prof. Andreas Dengel, for his continuous support and encouragement that has always motivated me to work hard and deliver the best results. I am also thankful to Prof. Sebastian Moeller for reviewing my thesis. I am thankful to all my colleagues in the MADM group. Thank you, Sebastian, Joern, Federico, and Stanislav, for the great discussions about diverse topics, both technical and non-technical. Thanks to Brigitte for being available for all the quick requests like meeting appointments in an otherwise busy schedule of Prof. Dengel. I am also thankful to Ms. Sabine Owens for patiently answering my queries and helping me complete many administrative tasks.

I am very thankful for the great opportunities of interning at Amazon Alexa AI and NVIDIA AI. The fruitful experiences introduced me to commercial NLP problems (thanks to discussions with colleagues at Amazon) and handling customer expectations (thanks to Dr. Thomas Breuel) for delivering higher standard results. I thank Prof. Lili Mou and Prof. Isabelle Augenstein for inviting me to their research labs and making me a part of their group. I really enjoyed discussions with colleagues at both labs, and learned a lot during the extensive meetings with Prof. Mou. Further, I am very thankful to the Ministry of Education and Research (BMBF) for recognizing my accomplishments and awarding me the AI Newcomer Award, which motivated me to continue doing good work.

I am very thankful to my husband, Shubham, who has been immensely encouraging all these years. Thanks, Shubham, for being supportive, understanding, and creating a healthy environment that kept me motivated despite difficult times like COVID. I thank my parents, Meenu and Ashwani, for teaching me the art of working hard and staying strong in every situation. Thanks, to my grandmothers, Mrs. Prem Lata Rikhi and Mrs. Kanta Jolly, for insisting on my education and showing me a gender-neutral society during my early years. I am also thankful to my brother, Devayansh, who, with his hysteric stories, nourishes perfect entertainment. I also feel grateful to my inlaws and my relatives for always counting me in their prayers.

At last, I am very thankful to all my school teachers, from Bhartiya Vidya Mandir, for recognizing my potential, protecting me from any distractions, and always insisting on the best scores I could achieve.

— Thank you, Shailza

CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACRONYMS

Seq2Seq	sequence-to-sequence
LXMERT	Learning Cross-Modality Encoder Representations from Transformers
METEOR	Metric for Evaluation for Translation with Explicit Ordering
PARENT	Precision And Recall of Entailed Ngrams from the Table
ROUGE	Recall Oriented Understudy for Gisting Evaluation
CIDEr	Consensus-based Image Description Evaluation
SBERT	Sentence-BERT
SOTA	state of the art
LSTM	Long-short Term Memory
NIST	National Institute of Standards and Technology
BUTD	Bottom-up and top-down attention
BLEU	Bilingual Evaluation Understudy
NLU	Natural Language Understanding
NLP	Natural Language Processing
NLG	Natural Language Generation
RNN	Recurrent Neural Networks
GRU	Gated Recurrent Units
CNN	Convolution Neural Networks
GAN	Generative Adversarial Networks
VQA	Visual Question Answering
GPT	Generative Pre-trained Transformer
CV	Computer Vision
ML	Machine Learning
DL	Deep Learning

Part I

INTRODUCTION

1 INTRODUCTION

The era of Deep Learning (DL) has led to substantial progress in multiple fields like object detection [135], image and speech recognition [130], neural machine translation [163] and many more. Deep models are leveraging vast computing resources and enormous training datasets to achieve the state of the art performances in numerous applications [5]. Among others, personal digital assistants like Apple's Siri[1], Amazon Alexa[2], and Google Assistant[3] by Google are successful deep learning applications. These assistants have become an integral part of our lives, helping us perform several daily tasks like booking a taxi, making appointments, and taking dictation at almost on-par human accuracy. NLP tasks like NLG and NLU are core components for building these assistants, and the advancements in these components directly translate into accuracy improvements.

advancements in deep learning translating to better user experience

However, the development of NLG and NLU systems generally requires large labeled datasets, which makes training these systems an expensive affair. The vast amount of training examples enable a model's learning such that it generalizes well at test times. However, we need to employ human annotators who annotate training pairs for training models, which makes this complete process expensive in terms of time, effort, and money. Usually, researchers use Amazon Mechanical Turk (AMT)[4] for data annotations and pay these annotators according to the difficulty of the task. Some tasks like VQA [7], an image conditioned NLU, employ ten annotators for each training sample which further increases the annotation cost.

data collection is an expensive affair

Therefore, data collection seems to be a bottleneck to designing DL applications, thus, limiting its use to mere big corporations. This limitation motivates this thesis in which we ask,

> Can we develop general methods to build data-constrained NLP systems?

In other words,

> Can we develop NLU and NLG systems with limited labeled data?

1 https://www.apple.com/siri/
2 https://developer.amazon.com/en-US/alexa
3 https://assistant.google.com/platforms/speakers/
4 https://www.mturk.com/

1.1 MOTIVATION

Data collection is an integral part of any Machine Learning (ML) or DL model's learning, but the expenses involved in the collection inhibit the development of new features. One can manage these costs by working with a data-constrained setting, a setting where we use limited training examples. There are multiple scenarios for a data-constrained setting, such as limited non-English training datasets, little task-specific datasets, etc.

limited training datasets for non-English languages

For the former, Bender [13] highlighted that most of the recent NLP research is based on 10-20 languages whose datasets are readily available, focussing primarily on English. Further, we have very limited datasets for threatened languages such as Yongning Na, a Sino-Tibetan language [1]. The language has only 40k speakers with only 3k written and unlabeled sentences. Therefore, due to the data-hungry nature of DL models, we can achieve high performance only for the high-resource languages. However, low-resource languages have performance issues because of the data shortage for low-resource languages [61].

limited task-specific datasets

In addition to the limitation of language-specific datasets, researchers face the problem of task-specific datasets. One such example is the work by Shah et al. [157], who showed that VQA models struggle with rephrasings at test time due to the absence of rephrasings in the training split of their dataset. Another real scenario would be the addition of new functionality in a dialogue agent [69], where we have minimal or no training data during the feature launch. Therefore, in a data-constrained setting, DL models show poor performance, making data collection a bottleneck to designing successful and scalable DL applications.

personal digital assistants emerging as friendly companions

In this work, we aim to build DL models in a data-constrained setting. We mainly focus on NLP, a vast application area of DL. Language technology is an emerging field with many real-world applications like dialogue agents. Users rely on dialogue agents, also known as personal digital assistants, to carry out various daily tasks and regard them as their emotional companions. In a study[5] from COVID times, users request Alexa to control their smart home appliances like televisions & air-purifiers around 860000 times per day. They spend most of their time with these agents to seek personal touch, like requesting Alexa crack a joke 9000 times a day in 2020 or asking her to laugh around 12000 times a day. Further, Alexa is optimized to process multiple modalities. One such example is a search on the Amazon Echo device, which returns visual responses to enhance the voice replies from Alexa[6].

5 https://www.mumbailive.com/en/tech/alexa-what-are-some-of-the-most-popular-questions-asked-by-indians-in-2020-61351

6 https://www.geekwire.com/2018/getty-images-visual-assets-will-enhance-alexa-searches-amazons-echo-devices-screen/

The above user queries and other interactions fall under the umbrella of NLU where these assistants understand the textual input, process it, and generate an answer (NLG) for the corresponding query. In this thesis, we aim to build these two components in a data-constrained setting. We focused on conditional NLU, where we studied the well-known task of VQA. We choose VQA due to its wide adoption in the research community, broad interest in industrial and commercial applications, and its implicit complexity that comes from integrating two different modalities; images and text.

A vast literature exists for handling data-constrained scenarios [61], such as data augmentation, distant supervision, data selection, exploiting feature representations, etc. These methods leverage additional sources or existing model artifacts and share the motivation to overcome the lack of labeled data. Here, data augmentation [34, 184] is one of the straightforward solutions when you only have limited labeled data with no unlabeled examples. Another scenario is the availability of small unlabeled corpora, where weak or distant supervision [179, 200] enables the model to learn by using both labeled and unlabeled datasets. Instead of using the entire training data, researchers have also explored data selection methods [93] which picks the most valuable examples for training the model while ensuring minimal or no accuracy losses.

methods to handle data-constrained settings

Despite their excellent results, data augmentation and distant supervision can only generate and extend task-specific training data. Therefore, they do not apply to scenarios when training and test distributions differ. Several works have explored methods of exploiting feature representations [35, 157] to handle such systems. Chapter 2 discusses each of these methods in more detail. We will explore some data-constrained settings scenarios and leverage the above-discussed dimensions to solve problems emerging from limited training data.

In the following sections, we summarize our research questions and the corresponding goals that we achieved as part of this thesis.

1.2 RESEARCH QUESTIONS AND GOALS

The main research question of this thesis is:
Q:

Can we develop NLU and NLG systems with limited labeled data?

The above question can be decomposed into five sub-questions with corresponding goals:

- **Q.1** :

Can we generate high-quality synthetic training examples with limited parallel data?

Goal: Check the feasibility of using DL models to generate paraphrases without parallel data. Design a model to generate high-quality synthetic data for bootstrapping new features with limited training data. an extensive evaluation of generated paraphrases so that they help the downstream tasks of intent classification and slot labeling.

- **Q.2** :

Can we minimize semantic information loss when training NLG systems with minimal training data?

Goal: Investigate few-shot data-to-text (a task of generating natural language description about table) generation. Investigate weak-supervision by leveraging the unlabeled tables. Design an unsupervised algorithm that minimizes semantic information loss in a few-shot setting. Provide quantitative and qualitative evaluation of the proposed model.

- **Q.3** :

Is "all you need is more training data" always true?

Goal: Analyze the already available ground-truth answers for each image-question pair. Use the entropy and semantic similarity of these answers to measure the informativeness of each example. Provide a quantitative study to show that we can recover a significant portion of model accuracy using a subset of the most informative training examples.

- **Q.4** :

Can we exploit feature embedding space to overcome dataset constraints?

Goal: Conduct empirical studies to understand the feature space of yes/no and non yes/no question types for VQA models. Present an extensive study of using yes/no questions to answer non yes/no ones (and vice-versa) and propose easier annotation methods for building future VQA datasets.

- **Q.5** :

Upto what extend transfer learning reduce data dependence in case of data distribution shifts?

Goal: Investigate the role of a language encoder in making VQA models robust to lexical variations. Empirically show that large pre-trained Transformer-based language models induce similar feature embeddings for the rephrasings by strongly emphasizing on keywords.

1.3 CONTRIBUTIONS

In this section, we discuss the already published contributions of this work. We summarize them into two broad areas: a) Methods for automatic data generation and selection, and b) Exploiting latent embedding space to overcome data constraints.

1.3.1 *Methods for automatic data generation and selection*

This part starts with the scenario when we only have limited labeled data. As discussed in Section 1.1, data augmentation is the most straightforward solution. We propose a novel method of data generation such that we can use the generated data to augment limited training examples.

1.3.1.1 *Data Augmentation*

We study a scenario of adding a new feature to a dialogue agent where minimal training data of this new feature is available. We design a neural-based paraphrase generation model that generates high-quality synthetic data. We introduce an interpretation-to-text model for paraphrase generation that uses existing training data for bootstrapping new features in task-oriented dialog systems. Our work is published in COLING 2020 [69].

interpretation-to-text model

1.3.1.2 *Improving Distant Supervision using Unsupervised NLG algorithm*

As an extension to the problem discussed in Section 1.3.1.1 where we have limited training examples from one intent, we study the data-constrained scenario where all classes in a labeled dataset have few instances. A model learned in such a few-shot setting suffers from a low semantic coverage problem, where important input table values are missing in the generated sentences. To overcome this problem, we use a small unlabeled dataset and employ distant supervision to generate sentence outputs for these unlabeled tables to augment limited training examples. However, the problem of *low-semantic coverage* still persists. Therefore, we develop a "search-and-learn" algorithm that

"search-and-learn" algorithm

exploits pre-trained language models (e.g., T5[143]) to fill the missing slots and improve the semantic coverage. Our work is published in AAAI 2022 [74].

1.3.1.3 *Inter-annotator agreement*

The above approaches use the entire training datasets, and depending on the availability of unlabeled data, we either augment it using data augmentation or use distant supervision. Using all training examples is justified when we have a few training examples. Nevertheless, during the availability of enormous amounts of training examples, we need to check if all the examples are helpful or not. Selecting the most informative subset of training data will allow us to train the models with minimum compute resources and considerably reduce the training times.

EaSe: a diagnostic tool based on answer diversity

We study the above scenario for VQA systems and propose a data selection tool that selects the most informative training examples to train VQA models. In VQA datasets, each image and question pair contains ten ground-truth answers from ten human annotators. We design a diagnostic tool called EaSe that leverages inter-annotator agreement of answers to classify the difficulty level of an image-question pair and select the smallest & most informative training examples. EaSe, based on **e**ntropy and **se**mantic similarity of annotator answers, is published in NAACL 2021 [73].

1.3.2 *Exploiting latent embedding space to handle data constraints*

The methods mentioned above extend and generate only task-specific training data. However, in real-world scenarios, one may face the problem of distribution shift where we can not use these methods. In the next part, we will discover such scenarios and exploit latent representations to overcome data constraints.

1.3.2.1 *Feature Space*

polar/non-polar feature space analysis

We study the widely used VQA 2.0 [56] dataset that contains 38% of questions with two answer classes (Polar questions: Yes/No) and the remaining ones spreading over 3,000 classes such as numbers & nouns (Non-Polar questions). We analyze the feature spaces of these two question types (Polar and Non-Polar). Our experiments show that we can answer non-polar questions using features induced by the model trained exclusively on polar questions referring to semantic concepts in non-polar questions (and vice-versa). Our work is published in ICPR 2020 [71].

1.3.2.2 *Transfer Learning*

We study the sensitivity of language encoders to question rephrasings in VQA models. One solution is to collect even larger datasets that account for these rephrasings. As a result, more time, human effort, and cost investments are needed to develop these models. Therefore, we a study showing that the use of large pre-trained Transformer-based language models can circumvent the requirement of additional data collection for building robust language encoders. Our work is published in EMNLP Findings 2020 [70].

vqa models robust to lexical variations

1.4 THESIS STRUCTURE

We have organized this thesis into five sections: Introduction, Background and Foundations, Methods for automatic data generation and selection, Exploiting latent embedding space to handle data constraints, and Conclusion and Future Work.

1.4.1 *Background and Foundations*

Chapter 2 presents the relevant background to understand the application of DL in NLP. We provide a detailed discussion about the two tasks, namely NLG and NLU. We also discussed some methods researchers use to design systems in data-constrained settings.

1.4.2 *Methods for automatic data generation & selection*

Chapter 3 introduces a scenario in which minimal training data of a new feature is available for a dialogue agent. A novel method of generating paraphrases, independent of any parallel data, is explained. The experimental results show that the proposed interpretation-to-text model improves the accuracy of downstream tasks of intent classification and slot labeling. The method obtains significant accuracy gains when applied to a commercial dialog system.

data augmentation

Chapter 4 presents an unsupervised algorithm for overcoming low semantic coverage problem in few-shot data to text systems. The proposed method recovers a majority of missing input table slots on several data-to-text datasets, largely alleviating the low coverage problem and closing the gap between few-shot and fully supervised learning.

unsupervised nlg algorithm

Chapter 5 presents leveraging inter-annotator agreements to select the most informative training examples. It explains the role of entropy and semantic similarity of annotator answers to choose the smallest and most informative training subset that recovers a significant portion of model accuracy.

inter-annotator agreement

1.4.3 *Exploiting latent embedding space to handle data constraints*

feature space

Chapter 6 presents the analysis of feature embedding space for VQA models. We conduct an extensive study of answering non-polar questions using polar-feature space if the polar questions used for training refer to the semantic concepts in the non-polar questions. The results indicate that polar questions, which are easier to annotate via crowd-sourcing, can be used to augment future VQA datasets.

transfer learning

Chapter 7 presents the limitation of current VQA models to generalize for lexical variations at test times. It shows the use of large pre-trained Transformer-based language models in inducing similar feature embeddings for the rephrasings by strongly emphasizing on keywords. The results establish that replacing the language encoder with a pre-trained model makes VQA models lexically robust.

1.4.4 *Conclusion and Future Work*

Chapter 8 presents the conclusion and Chapter 9 explains about some promising future directions.

Part II

BACKGROUND AND FOUNDATIONS

2 BACKGROUND AND FOUNDATIONS

In this chapter, we provide the background about DL in the field of NLP. The area of NLP has progressed from traditional ML models to recent Transformer-based models that have sparked progress in various applications. Among other successful NLP applications, digital personal assistants have gained huge popularity (Section 1.1), and the development of NLG and NLU components explains their good performance. Therefore, this chapter intends to introduce NLG and NLU architectures to the reader and make them aware of the state of the art (SOTA) in this field of research.

chapter structure

We organize this chapter as follows: Section 2.1 explains the deep learning for NLP. We demonstrate the progress of NLP by discussing both traditional and current SOTA approaches. This section explains important deep learning architectures namely Convolution Neural Networks (CNN) (Section 2.1.1), Recurrent Neural Networks (RNN) (Section 2.1.2) and Transformers (Section 2.1.3). We will then discuss about NLG in Section 2.2. This section will mainly focus on traditional approaches for data-to-text generation; the main focus for this work. We conclude this section by providing a brief overview of the modeling approaches, the evaluation, and common datasets for data-to-text systems. Section 2.3 describes NLU and will introduce multimodal-NLU i.e., the task of image-conditioned question answering (VQA). We will discuss VQA datasets, their modeling approaches, and their evaluation metric. After explaining these components, we will discuss data-constrained NLP in Section 2.4. Finally, we will conclude this chapter with a summary in Section 2.5.

2.1 DEEP LEARNING FOR NLP

DL is a subfield of ML to understand the nature of human (and other forms of) learning and build learning capabilities in computers. Similarly, Neural Network is also a subfield of ML and is the backbone of deep learning algorithms [55]. Neural networks [119] mimic the human brain and are inspired by the process in which biological neurons signal to one another. These networks are comprised of an input layer, output layer, and one or more hidden layers and are the core behind the deep learning models, where the word "deep" refers to the depth of layers in a neural network. These networks are known for their representational capabilities and have shown great success in various fields like natural language processing [193], computer vision [177], healthcare [126], etc.

NLP deals with building computational algorithms for the automatic examination and representation of human language. These algorithms aims to teach computers the ability to perform complex natural language-related tasks like question answering, document understanding, machine translation, etc. The immense growth of NLP has allowed the release of many successful applications like Google's search engine[1], Amazon's Alexa[2], Grammarly[3], etc., which many users use for the easiness of their daily lives.

traditional nlp

Before the recent approaches, most methods employed shallow ML models, like logistic regression [33] and SVM [31], to solve NLP problems. These traditional approaches trained on very high dimensional and sparse features, leading to issues like the curse of dimensionality. Furthermore, traditional approaches used hand-crafted features, making the process both time-consuming and expensive. Neural networks handled some of these limitations by introducing the distributed representations of word [14] and enabling an automatic feature representation learning.

word2vec

In 2013, Mikolov et al. [123] released two models to construct high-quality distributed word representations, also known as word2vec, which eventually sparked the progress in deep learning for NLP. The two neural-based models, CBOW and skip-gram, are approaches to constructing word embeddings. The objective for CBOW is to compute the conditional probability of a target word given the context words in a given window size. For skip-gram, the goal is to predict the conditional probability of context words given a central target word. The intuition behind word2vec is that words with similar meanings tend to occur in a similar context, and therefore similar words will have similar vector representations. This way, the neural approach encoded the word semantics within the word representations. For both models, the dimension of word embeddings is determined by computing the prediction accuracy. Researchers used these pre-trained word representations to encode input sequences for multiple tasks like sentence classification, sentiment analysis, etc.

Despite achieving good performances on multiple NLP tasks, these vector representations hold some limitations. Their window-based approach considers only a few words since the model size will increase with more oversized windows. The use of smaller window sizes produces similar embeddings for contrasting words like "bad" and "good" [164], which is not desirable for tasks like sentiment analysis [181] where differentiation plays an important role. Additionally, the fixed window neural model limits the context, which leads to incorrect predictions. These limitations motivated the introduction of

1 https://www.google.de/?hl=de
2 https://developer.amazon.com/en-US/alexa
3 https://www.grammarly.com/

two crucial neural architectures, namely CNN and RNN, that processed inputs of any length without increasing the model size.

CNN and RNN are the two basic network architectures that powered the progress in deep learning. CNN is mainly used for image processing, and RNN is used for NLP. Since the introduction of these basic architectures, researchers have worked on their advancements and have applied these networks to solve multiple tasks like text classification [104], image recognition [158], etc. We briefly introduce these architectures and discuss their variants in the following sections.

2.1.1 *Convolution Neural Networks*

A CNN is a deep learning architecture designed to analyze visual images. CNN performs successive convolution operations over the images to capture their spatial representations. CNN consists of two main components feature learning and output layer depending on downstream tasks. The earlier layers of CNN perform feature learning by learning low-level features like edges and corners, such that higher layers use these features to classify the input image. Each convolution layer uses the output from the preceding layers and learns to combine these low-level features with learning a higher level of abstraction. Like traditional neural networks, the outputs from each layer are passed through non-linear activation functions like sigmoid, tanh, or relu. These activation functions increase model expressiveness and prevent the values from exploding. CNN also have pooling layers to reduce the spatial dimensions of intermediate image representations.

convolution neural networks

After feature learning, the convolution matrices are flattened into a vector, and fully-connected layers are added to the network for final classification. For classification task, the final layer is a softmax layer, which converts the activations into a probability distribution, s.t., all values add up to 1, over the label space.

For many computer vision tasks, these CNN have turned out to be a natural choice [66, 87, 158]. One of the first researchers to apply CNN based frameworks to NLP problems was Collobert and Weston [29], who used a look-up table to transform words into a vector representation and applied 1-D convolutional filters to produce a feature map. Following, they applied a max-pooling operation to obtain the final sentence representation. In a follow-up study by Collobert et al. [30], they proposed a general CNN-based framework that solves many NLP tasks. Despite the great success of CNN in the succeeding literature [78, 198], the models hold some shortcomings.

CNN are data extensive models, i.e., they need substantial training examples due to many trainable parameters [193]. Further, CNN struggle to model long-distance contextual information and fail to preserve the sequential order in their representations [65, 78]. RNN, discussed in Section 2.1.2, address these limitations and are designed to model sequential information for creating final sentence representations.

2.1.2 *Recurrent Neural Networks*

RNN [46] is a deep learning architecture designed for processing sequential data. A sequential data cannot see future events and takes past states into account while processing the current state. Common types of sequential data are natural language and time series. Unlike weights in traditional neural networks that don't capture the notion of similarity between input words, RNN have weights shared at all time steps. RNN work well for shorter sequences, but they fail to capture long-term dependencies for longer sequences. The main challenge for RNN is the vanishing gradient problem [62]. A vanishing gradient problem refers to a case when gradients during backpropagating become small due to repetitive multiplications and, therefore, it makes RNN incapable of learning long-term dependencies.

recurrent neural networks

This shortcoming of RNN leads to the introduction of Long-short Term Memory (LSTM) [63]. LSTM overcome the vanishing gradient problem by introducing different gates and a cell state (c). The cell state stores long-term information, and multiple gates control the erasing, writing, and reading of information from the cell. The three gates of LSTM are forget gate, input gate, and output gate. The three gates are :

long short-term memory cells

$$i_t = \sigma(w_i[h_{t-1}, x_t] + b_i), \tag{2.1}$$
$$f_t = \sigma(w_f[h_{t-1}, x_t] + b_f), \tag{2.2}$$
$$o_t = \sigma(w_o[h_{t-1}, x_t] + b_o), \tag{2.3}$$

Here, i_t, f_t and o_t represent the input, output and forget gate. It uses the sigmoid activation function, which clamps values between 0 and 1. A value of 0 for forget gate encourages the network to not forget anything, thus allowing LSTM to preserve information for longer times. Each gate has its own sets of weights, and biases like w_f, b_f are weights and biases for forget gate. x_t refers to the input at the current timestep, and h_{t-1} refers to the previous hidden state input.

We use these gates to decide the output of LSTM. The cell state equations are:

$$c'_t = \tanh(w_c[h_{t-1}, x_t] + b_c), \tag{2.4}$$
$$c_t = f_t * c_{t-1} + i_t * c'_t, \tag{2.5}$$
$$h_t = o_t * \tanh(c_t), \tag{2.6}$$

here c'_t, c_t are the intermediate cell state and cell state respectively. h_t refers to the hidden state at time step t, which is passed to the next time step as input.

Gated Recurrent Units (GRU) [27] is another RNN variant. GRU combines forget and input gates into a single update gate. It also merges cell state and hidden state into one single state. GRU also handle vanishing gradient problems and are widely used because of their small size (fewer parameters).

gated recurrent units

Researchers showed the success of RNN in various applications like Lample et al. [90] proposed to use bidirectional LSTM for NER tasks, Wang et al. [181] proposed the encoding of entire tweets using LSTM and used the hidden states for predicting sentiment polarity. RNN has also been used in challenging applications like generating natural language (NLG) and conditional language understanding (NLU). Sutskever, Vinyals, and Le [169] proposed a general deep LSTM encoder-decoder framework that maps one sequence to another. Here, the encoder is an LSTM that encodes the "source" sequence as a fixed-size vector, and the vector is used as an initial hidden state for another LSTM called decoder. Here, the sequence on the encoder side is task-specific, like text in the original language for machine translation, the question to be answered for question answering, and the table to be summarized for tabular summarization. The encoder-decoder network is trained end to end using backpropagation through time [186]. The decoder generates one token at every time step and updates the hidden state with the last generated token during inference. Vinyals and Le [175] employed the same encoder-decoder framework to model human conversations. Researchers also explored conditioning on visual inputs inspired by conditioning on language inputs. One such task is image captioning [176], which is considered a translation problem with the only difference across multiple modalities, namely from visual to text modality. Visual Question Answering [7] is another example of an image-conditioned question answering task.

applications of seq2seq architectures

The traditional encoder-decoder frameworks force the last hidden state to encode the entire input information, increasing the difficulty of modeling long-term dependencies. Bahdanau, Cho, and Bengio [11] reported this problem for machine translation and applied attention mechanisms which showed good performance, especially for long sequences. The attention mechanism uses a direct connection to the encoder to focus on specific parts of the source sequence on each decoder step. In other words, attention is a technique to compute the weighted sum of values (encoder hidden states) dependent on the query (decoder hidden state). Motivated by the success of attention to improve translation systems, researchers have used attention in CNN and RNN based models which improved performance for various tasks like summarization [150], sentiment analysis [182], image captioning [190], etc.

attention mechanism

However, CNN and RNN fail to robustly model long sequences and suffer from the sequential processing during the encoding of the in-

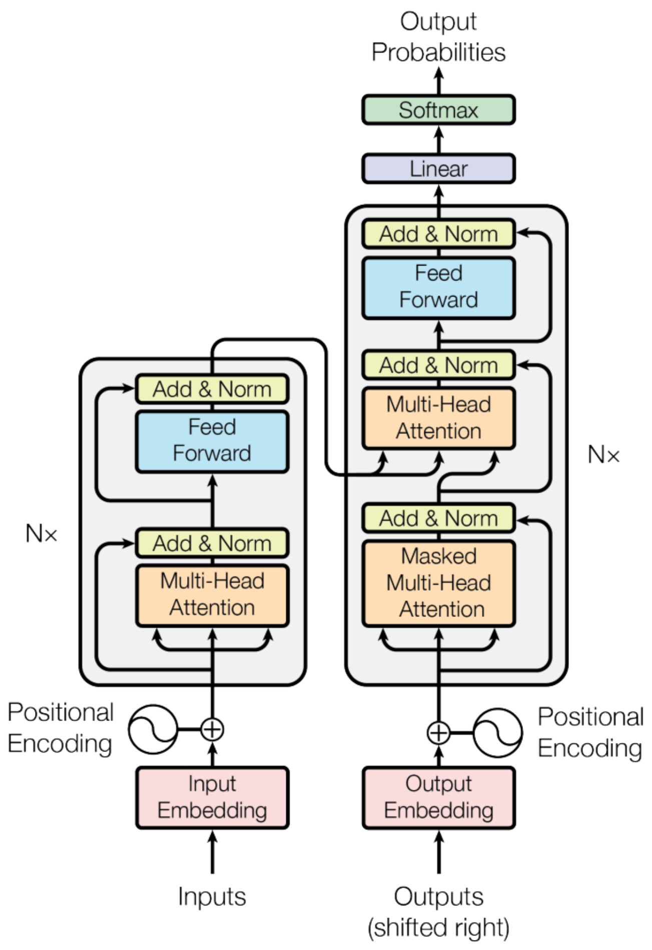


Figure 2.1: Architecture for the Transformer model (Image Source: [173]).

put sequence. Furthermore, the recurrence behavior of RNN makes them extremely slow and is not parallelizable. Vaswani et al. [173] addressed these problems in the Transformer model that uses only attention to capture the relationships between input and output. We will discuss the Transformer model in the following sections.

2.1.3 *Transformers*

Transformers by Vaswani et al. [173] is a novel architecture that relies entirely on self-attention to compute input and output representations. Each word attends to its neighboring words to encode its representation. Unlike RNN or CNN, Transformers are fast, parallelizable and capture the long-term dependencies. Transformers have shown huge improvements in various NLP tasks, especially for pre-trained models [138]. They have been widely adopted in other fields like computer vision [16, 41], audio processing [21, 39], etc.

Figure 2.1 shows the Transformer architecture, a sequence-to-sequence (Seq2Seq) model [169] with a stack of identical encoders and decoders.

As mentioned in Section 2.1.2, attention is a technique to compute the weighted sum of values dependent on the query. Instead of applying a single attention function, the Transformer uses multi-head attention, i.e., using different sets of learned projections to project original queries, keys, and values and add them to the desired dimensional representation. This way, each attention head learns different language characteristics like semantics, syntax, etc., to produce rich representations at the end. Each encoder block includes multi-head self-attention followed by a layer of the feed-forward network. In addition to these two sub-layers, the decoder contains multi-head attention over the output of the encoder stack. Transformers ensure that the outputs don't see the future on the decoder side; therefore, future events are masked on the decoder side. The architecture comprises residual connections [60] and layer normalization [10] for efficient training.

transformer architecture

Transformers architectures have considerable representation capacities, and therefore researchers leveraged these architectures to create multiple foundational models in which they pre-trained the encoder, decoder, or both. One can finetune these pre-trained models with limited training examples to achieve SOTA results on various NLP tasks.

Devlin et al. [35] released a **pre-trained Transformer encoder** on two pre-training tasks: masked language model and next sentence prediction. The proposed model is **BERT**, which stands for **B**idirectional **E**ncoder **R**epresentations from **T**ransformers. BERT pre-trains deep bidirectional representations from the unlabeled text by jointly conditioning on both left and right contexts in all layers. Due to the consideration of bidirectionality in BERT, it is an ideal choice for tasks where we need to encode a complete sequence, such as for the tasks like sentiment classification [128], question answering [140], etc. RoBERTa [108] and ALBERT [91] are further extensions with improvements over the pre-training data and methodologies like parameter-reduction techniques [91].

transformer based foundational models

Another foundational model called Generative Pre-trained Transformer (GPT) by Radford et al. [141] is based on a pre-trained Transformer decoder. The name "generative pre-training" suggests using text generation as the pre-training objective. GPT is an autoregressive language model trained using BooksCorpus dataset [202] which contains over 7000 unpublished books from various genres. The pre-training of GPT on long stretches of contiguous text allows the model to learn to condition on long-range information, thus making it ideal for various NLP tasks. GPT-2 [142] and GPT-3 [15] are further extensions with improvements on using larger dataset and adding more parameters.

Following the separate pre-training of encoder and decoder blocks, researchers also released encoder-decoder-based models pre-trained on Seq2Seq based tasks [96, 143]. T5 by Raffel et al. [143] is one of the widely used model that treats every text processing problem as text-to-text and has achieved good performance on a wide variety of English-based NLP problems [143]. The success attributes to the large pre-training dataset, which is large, diverse, and clean, making it an ideal choice for several pre-training tasks.

This completes our discussion about the progress of DL in NLP. In the following sections, we will discuss in detail the two NLP tasks, namely NLG and NLU.

2.2 NATURAL LANGUAGE GENERATION

NLG is the task of generating natural language output such that it resembles a human writing style. Defining NLG is difficult [47] since the output of an NLG system should be text, but the exact inputs can vary substantially [120]. There are multiple applications of NLG like:

applications of nlg

1. Machine Translation [132, 165]; a task of translating (generating) sentence from one language to another,
2. Question Answering [17]; a task of generating answer for a given question based on the provided context,
3. Simplification [161]; simplify sentence so that it is easily readable even by readers with basic education,
4. Summarization [28]; combining sentences to make them concise for better readability,
5. Dialogue Response Generation [19]; a task of generating dialogue agent's response

2.2.1 *Data-to-text Generation*

In this thesis, we have focused on NLG for dialogue systems. Dialogue systems have input data in a structured form or a table, i.e., having key-value pairs, also known as slot names and slot values. The task of NLG is to summarize those key-value pairs in a natural language sentence. Researchers address this problem as data-to-text generation and have released data-to-text datasets for multiple applications like restaurant descriptions [131], biographies [95], weather forecasts [101], etc. In the following sections, we will explain recent architectures for the data-to-text generation, benchmarking datasets, and their evaluation metrics.

2.2.2 *Modeling approaches for Data-to-text Generation*

traditional approaches to nlg

The traditional approaches to NLG [92, 149, 166] use hand-crafted rules with statistics that lack flexibility and diversity of generated sentences. Generation comprises modular and independent decisions like (1) content selection decides the selection of parts of the input field, (2) sentence planning decides the presence of selected parts in the output sentence, (3) surface realization generates sentences. Works like Reiter and Dale [146] follows a pipeline approach of content planning and surface realization, while some works use hand-engineered rules [88] and statistical induction [101]. However, these approaches suffer because of the flexibility, diversity of generated sentences and error propagation between pipeline stages.

Neural models consider this limitation and leverage massive amounts of parallel data for training the text generator [95, 106, 155, 187]. These models combine all three modules of the pipeline approach and learn a single model end-to-end that determines the relevant input records, dependencies between them, and the best way to describe them. One of the baseline text generation model is an encoder-decoder architecture, a Seq2Seq model [169], which is inspired by the advances in machine translation. .

neural approaches to nlg

As mentioned in Section 2.1, Seq2Seq model by Sutskever, Vinyals, and Le [169] uses two LSTM networks. The first LSTM network encodes input sentence in the source language to an n-dimensional vector, and the second LSTM uses this n-dimensional vector to decode it into the sentence in the target language. Data-to-text can also be considered as a translation problem, where we will use encoder LSTM to encode our data (table), and the decoder will generate a natural language description of the table. Tabular data (T) is a structured input, a set of slot name–value pairs, denoted by $\mathbf{T} = \{(s_i, v_i)\}_{i=1}^{S}$, where s_i is the name of the ith slot, v_i is the value, and S is the number of slots. The output is a sentence $\mathbf{y} = (y_1, y_2, \cdots, y_n)$ that describes the given input $\mathbf{T}$.

neural approach for data-to-text

One of the first works by Mei, Bansal, and Walter [122] uses LSTM-RNN to select and generate natural language descriptions. A bidirectional LSTM encodes input records, a novel coarse-to-fine aligner reasons over the input for deciding the selection of records, and finally, an LSTM decoder generates a natural language description of selected records. Other works also follow the same technique with slight variations [95, 101, 106]. Recently, pre-trained Transformer [173] based models have shown great success in the field of language generation [15, 143].

2.2.3 *Datasets for Data-to-text Generation*

Researchers use high-quality datasets to evaluate the performance of their modeling approaches. There are many datasets for data-to-text generation like E2E [131] from restaurant domain, WikiBio [95] from biography domain, WebNLG [51] contains triplesets from DBPedia [9], ToTTo [134] designed for controlled data to text generation. We

data-to-text datasets

will explain the two datasets that we used in our experiments described in Chapter 4.

e2e

E2E is a crowd-sourced dataset from the restaurant domain and contains more than 50K table–text pairs. The input table contains slot names and slot values like slot value "yes" for slot name "family-friendly" indicates that the restaurant is family-friendly. There are 3-8 slots in the input, and the output reference contains one or a few sentences. The dataset is available for download at `http://www.macs.hw.ac.uk/InteractionLab/E2E/`.

wikibio

WikiBio contains 700K biographies from Wikipedia along with a tabular infobox. It treats the first sentence of the article as a reference for each biography. The dataset is available for download at `https://github.com/DavidGrangier/wikipedia-biography-dataset`

2.2.4 *Evaluation of generated texts*

evaluation of nlg

Evaluation of NLG systems is a challenging task, and researchers employ both qualitative and quantitative measures to evaluate machine-generated outputs. The quantitative evaluations involve automatic metrics, discussed in Section 2.2.4.1, and the qualitative evaluation employs evaluation by humans (Section 2.2.4.2).

2.2.4.1 *Automatic Evaluation*

automatic evaluation

Automatic metrics compare the machine-generated response with ground truth references, where these references are obtained from human annotators. Metrics like Bilingual Evaluation Understudy (BLEU) [133] consider n-gram overlaps between the generated and ground-truth reference. The metric has been widely adopted but has some important limitations, like a semantically equivalent sentence with low n-gram overlap with ground-truth gets a low BLEU score. Metric for Evaluation for Translation with Explicit Ordering (METEOR) [94] extends the idea behind BLEU by introducing the use of synonyms while calculating the score. Recall Oriented Understudy for Gisting Evaluation (ROUGE) [102] is a recall-oriented metric that performs an n-gram recall over ground truth text. National Institute of Standards and Technology (NIST) [38] is another metric based on BLEU. In contrast to BLEU, which gives equal weights to each n-gram, NIST calculates the score by giving more weight to rare words. Consensus-based Image Description Evaluation (CIDEr) [174] metric computes Term Frequency Inverse Document Frequency (TF-IDF) weighting for n-grams.

parent metrics for data-to-text generation

For a data-to-text generation, Dhingra et al. [36] observe that BLEU does not correlate well to human satisfaction. They propose Precision And Recall of Entailed Ngrams from the Table (PARENT) metrics that is computed against both input table and a ground-truth reference. In their study, they show a high correlation between PARENT scores and the human judgment, thus making PARENT ideal for the data-to-text generation.

All these metrics have their advantages and disadvantages, hence making them inapplicable for some scenarios. Therefore, researchers design task specific criteria to ensure the correctness of generated output. In Chapter 3 and Chapter 4 we will present some the metrics that we designed and used for evaluating the correctness of our approach.

2.2.4.2 *Human Evaluation*

human evaluation

In addition to automatic metrics, qualitative evaluation is also performed by employing human annotators. However, it is an expensive process, and we need to provide clear task descriptions to avoid subjective biases from annotators. Human evaluation can be done using crowd-sourcing platforms like Amazon Mechanical Turk[4], and CrowdFlower[5]. Furthermore, some works select a random subset of samples and employ at least three annotators to evaluate task-specific criteria [8].

2.3 NATURAL LANGUAGE UNDERSTANDING

NLU is a subfield of NLP which aims to make machines understand the natural language by deriving the semantics, identifying the context, and interpreting language correctly. Developing NLU components enable machines to perform complex tasks like word sense disambiguation [42]. Word sense disambiguation is a task in which models understand two different senses of the word "bank" in the sentences; "Our banks are closed on Thursday" and "This year, due to floods, the river will overflow the banks". NLU is essential for many other applications like sentiment classification [128], question answering [140], etc. Even the task of data-to-text generation incorporates the NLU component, which aims to understand the intent behind the text in the tabular form to generate human-like natural language text.

2.3.1 *Visual Question Answering*

In this thesis, we focused on conditional NLU, where we studied the well-known task of VQA. We choose VQA due to its broad adoption in the research community, wide interest in industrial and commercial applications, and its implicit complexity that comes from integrating two different modalities; images and text.

VQA is an image-conditioned NLU task or image-conditioned question answering task, where similar to the question answering systems, VQA models provide an answer based on the given image. VQA is a

4 https://www.mturk.com/
5 http://faircrowd.work/platform/crowdflower//

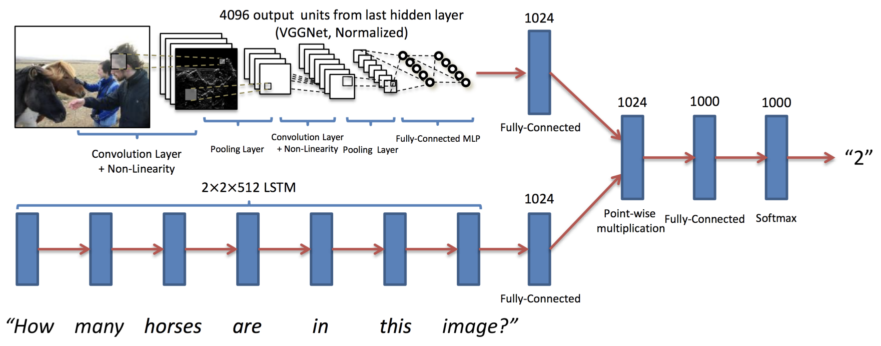

4096 output units from last hidden layer (VGGNet, Normalized)
Convolution Layer + Non-Linearity
Pooling Layer
Convolution Layer + Non-Linearity
Pooling Layer
Fully-Connected MLP
1024
Fully-Connected
1024
1000
1000
"2"
Point-wise multiplication
Fully-Connected
Softmax
2×2×512 LSTM
1024
Fully-Connected
"How many horses are in this image?"

Figure 2.2: Baseline VQA model (Image Source: [7]).

multimodal task using separate encoders for each modality. In the following sections, we will discuss the baseline and current models in the field of VQA.

2.3.2 *Modeling Approaches for VQA*

baseline vqa model

Antol et al. [7] introduced this task and released a first model as shown in Figure 2.2. The model uses CNN variant VGGNet [162] for encoding images and an LSTM to encode questions. Each of these encoded vectors is followed by a softmax over K possible outputs. Antol et al. [7] choose 1000 top answers as possible outputs covering almost 80% of answers in the training and validation splits. Among all the models mentioned in [7], the baseline model achieved maximum accuracy of 57.75% for the open-ended task of VQA.

other vqa models

The introduction of VQA led to the emergence of multiple models like Stacked Attention Networks for Image Question Answering [192] that uses separate attention module inspired from the works of image captioning [190] and machine translation [11]. Xu and Saenko [189] introduce spatial Memory Network for the VQA task. The above models focus only on visual attention [188, 189, 192] while works like [112] presented a novel co-attention model for VQA that jointly reasons about both image and question attention.

winner of 2017 vqa challenge

Anderson et al. [6] proposed one of the most used and accepted VQA models called Bottom-up and top-down attention (BUTD). BUTD use bottom-up & top-down attention for image captioning and VQA tasks, and became the winner of the 2017 VQA challenge. BUTD uses a GRU to encode the input questions and attends the image ROI features to enable region-based attention for answer generation.

Recent research in VQA, inspired by the Transformers [173], focuses on pre-trained Vision and Language models. Vision & Language pre-trained networks, like LXMERT [170], VilBERT [111], VLP [199], have shown great success in the Visual Question Answering field. Each of these models is Transformer-based and employs BERT [35] like pre-training objectives. Learning Cross-Modality Encoder Representations from Transformers (LXMERT) is a Transformer-based pre-trained model. LXMERT focuses on vision-language interactions to enable a better understanding of each modality and the relationship between them. It contains separate encoders like an object relationship encoder, a language encoder, and a cross-modality encoder. LXMERT uses five different vision-language tasks like masked cross-modality language modeling, cross-modality matching for pre-training. Similar to this, ViLBERT is also pre-trained on diverse downstream tasks, including VQA, referring expression, and image-to-text retrieval. VLP by [199] propose a unified encoder-decoder model for general vision-language pre-training.

transformer based pre-trained vision & language models

2.3.3 *VQA Datasets*

Following the discussion of top modeling approaches in VQA, we will discuss some important datasets that researchers use for the evaluation of their modeling approaches. Researchers introduced a wide range of VQA datasets. Antol et al. [7] introduced one of the first datasets for VQA, VQA 1.0, in which they employed ten human annotators to answer each image and question pair. The dataset contains over 760K questions with around 10M answers. Despite being large, VQA 1.0 is not used because of its linguistic biases.

vqa1.0

Goyal et al. [56] handled this problem by releasing a large and balanced VQA 2.0 dataset. VQA 2.0 contains complementary images s.t., a pair of similar images result in two different answers for the same question. The dataset ensures that VQA models learn visual grounding for answering a given question rather than relying on dataset biases. There are 443757 and 214354 training & validation samples in VQA 2.0. VQA 2.0 is one of the widely used datasets in the VQA community. VizWiz is another important dataset that comes from a natural VQA setting. Gurari et al. [59] released the first goal-oriented VQA dataset called VizWiz. Unlike all the above datasets, VizWiz originates from a natural VQA setting. Blind people use their mobile phones to click images and record spoken questions about them. This real-world scenario makes this dataset quite challenging and contains unanswerable questions. The dataset is considered one of the most challenging datasets containing blurred images and conversational questions. There are 31,000 visual questions and ten crowdsourced answers for each question.

two widely used vqa datasets

The other VQA datasets like DAQUAR [117], COCO-QA [147], Visual madlibs [194] have faded away from use and, therefore, are not considered as benchmarks in the related work. We will discuss the evaluation metric for VQA systems in Section 2.3.4.

2.3.4 *Evaluation of VQA systems*

evaluation of vqa

Researchers release a wide range of VQA metrics like Arithmetic and Harmonic Means [76], WUPS [116], MaSSeS [72] but the most commonly used is Accuracy [7, 56, 59, 196]. Since VQA is a classification task, it counts the matches between the model's prediction and the original ground truth answer. Our VQA model predicts one ground truth answer, and we have ten ground-truth answers for datasets like VQA 2.0, VQA1.0, VQA-Abstract, and VizWiz.

$$\text{ACC} = \min\left(\frac{\text{humans that said answer}}{3}, 1\right) \tag{2.7}$$

standard vqa metric

While Accuracy works well for a single ground-truth answer, [7] designed a special VQA metric that takes into account multiple ground-truth answers. Equation 2.7 refers to the standard evaluation metric of VQA systems. It averages over all ten over nine sets of ground-truth answers. The metric is based on the majority vote, i.e., a predicted answer answered by at least four annotators gets 100% accuracy, and for three, two, or one human vote, the accuracy values are 90%, 60%, 30%, and 0%.

This completes our discussion about two crucial instances of NLP, namely NLG and NLU, where these models leverage tremendous amounts of training examples to achieve better results for their in-hand downstream tasks. However, as explained in Chapter 1, data collection is an expensive process and, therefore, the scope of successful NLP applications is limited to big corporations. In this thesis, we are working towards data-constrained settings. We will discuss some background about data-constrained NLP in the following sections.

2.4 DATA-CONSTRAINED NLP

Data-constrained is a setting where we have limited or no training examples. Researchers have addressed data-constrained, also known as low-resource, scenarios in various domains like limited datasets for threatened languages [1]. Bender [13] presented a study in which they showed that most of the recent NLP research is based on 10-20 languages whose datasets are readily available. In addition to the limitation of language-specific datasets, researchers face the problem of task-specific datasets. One such example is the work by Shah et al. [157], who showed that VQA models struggle with rephrasings at test time due to the absence of rephrasings in the training split of their dataset.

Researchers have proposed standard methodologies for data-constrained scenarios, where all the methods leverage additional sources or existing model artifacts and share the motivation to overcome the lack of labeled data. Additional sources like unlabeled data, manual heuristics, or cross-lingual alignments may vary depending on the target task. One needs to understand the requirements of these methods for choosing a technique well suited for the low-resource setting. Hedderich et al. [61] presented an extensive review highlighting the underlying assumptions for different techniques in a low-resource setting. Following, we will discuss some of these methods.

One of the most simple data-constraint scenarios is when you only have limited labeled data and no additional datasets like unlabeled ones. **Data augmentation** is one of the most straightforward solutions to handle this scenario. It's the method to obtain new instances based on the existing ones by modifying the features with transformations that don't change the label. Data augmentation has been a popular approach in Computer Vision (CV) [160], e.g., an image rotation doesn't change image content classification. In NLP, researchers have adopted data augmentation by replacing words with their synonyms [184], or entities of the same type [34]. Some works perform data augmentation at the sentence level like Ma et al. [114] used back-translation for the data-to-text generation.

data augmentation

Another scenario is the availability of small unlabeled corpora in addition to the limited labeled training examples. **Distant or weak supervision** can handle this scenario, where we can obtain the corresponding labels through a (semi)-automatic process from an external information source. Mintz et al. [125] introduced distant supervision for relation extraction with extensions on multi-label learning [168] and multi-instance [148]. Another work by Wang et al. [179] transfers a document-level sentiment label to all its sentence-level instances. Self-training method or semi-supervised learning is also considered weak supervision. Zhu and Goldberg [200] introduced semi-supervised learning, a learning paradigm that uses unlabeled data to improve supervised learning tasks in data-constrained settings. This enables DL models to use both labeled and unlabeled datasets.

weak supervision

The above approaches use the entire training data and augment it using either data augmentation or weak supervision, depending on the availability of unlabeled data. The use of whole training data is justified when the number of examples is scarce. However, during the availability of enormous amounts of training examples, one must consider if all the training examples are helpful or not. This way, selecting the most informative subset of training data will allow us to train the models with minimum compute resources and considerably reduce the training times.

Researchers have addressed this problem and have designed **data selection** mechanisms to pick the most useful training examples for training. A study by Lapedriza et al. [93] showed that object detection and classification systems don't benefit from all training examples. They designed an approach to measuring the training value of an example and improved SOTA detectors and classifiers by using the

data selection

training subset instead of whole training data. Some other works in the same field of research have also shown the effectiveness of using training subsets over the entire training datasets [25, 53, 85, 151]. Hence, one can leverage data selection methods in designing better datasets with a few high-quality training examples instead of many low-quality training examples, thus making data collection possible in a low-resource setting.

exploiting feature representations

The above approaches work assuming that training and test datasets have a similar distribution. However, this is not true in a real-world scenario, as users can ask questions in different forms. Researchers have addressed this problem by releasing datasets like Agrawal et al. [3] released new test splits and showed that the performance of VQA degrades significantly when test and training distributions are different. Similarly, [157] showed that these models fail when questions are replaced by their rephrasings at test time. Data augmentation and distant supervision are not applicable in such scenarios as these methods generate and extend only task-specific training data. A ML method called **Transfer Learning** handles such problems. It reduces the need for labeled target data by transforming models and learned representations. In NLP, recent works on transfer learning use pre-trained language representations induced by pre-trained models, like BERT [35] and GPT [141], trained on large unlabeled text corpora.

As discussed in Chapter 1, this thesis mainly focuses on digital personal assistants, a successful NLP application, where these assistants perform tasks related to NLG and NLU. We aim to build NLG and NLU components in a data-constrained setting. For the rest of the thesis, we will explore some data-constrained settings scenarios and leverage the dimensions discussed in Section 2.4 to solve problems emerging from limited training data.

2.5 SUMMARY

chapter conclusion

We started this chapter with a detailed discussion about the progress of DL in NLP. We provided a detailed discussion about DL models followed by two important applications of NLP. We discussed a brief overview of data-to-text generation, a sub-task of NLG, its evolvement from traditional template-based approaches to the recent neural-based encoder-decoder models. We discussed the role of pre-trained Transformer models in natural language generation. The major datasets for data-to-text generation, namely E2E and WikiBio, were also discussed. Finally, we discuss quantitative and qualitative evaluations of NLG systems. We also discussed Visual Question Answering, an image-conditioned NLU task. A brief discussion of neural-based models and recent Transformer-based pre-trained models is presented. We discussed mostly-used benchmarking VQA datasets and concluded this chapter with the methods addressing data-constrained NLP.

Part III

METHODS FOR AUTOMATIC DATA GENERATION & SELECTION

3 DATA AUGMENTATION

In this chapter, we study one of the most straightforward scenarios when we only have minimal labeled training data. This scenario arises when adding new functionality for a task-oriented dialogue agent, for which we have very few training examples. We present the proposed methods and models to handle the limited training data using data augmentation, one of the methods described in Section 2.4.

Dialogue agents answer multiple user queries after being trained from large-scale training utterances from different domains like music, weather, and navigation. However, we have minimal training data when adding a new feature. We can use paraphrasing models to generate more examples for the limited training data; however, these paraphrasing models need parallel datasets for training which is again an expensive and time-consuming process in terms of data collection. Therefore, this chapter proposes an *interpretation-to-text* model for paraphrase generation that uses existing training data for bootstrapping new features, which improves the accuracy of downstream tasks of intent classification and slot labeling. We run our experiments on a public dataset in English & a commercial dialogue system, real-life data in the German language, and we observe improvements for downstream tasks, demonstrating the usefulness of our approach.

publication and chapter structure

The methods and models presented in this chapter have been published in COLING 2020 [69]. The rest of the paper is organized as follows. Section 3.1 explains the problem definition. Section 3.2 explains the related work for data augmentation and paraphrase generation. Section 3.3 explains our data augmentation approach. We explain our proposed *interpretation-to-text* model for paraphrase generation. The section also explains different paraphrase sampling strategies and the mechanism of label projection. Section 3.4 explains the two datasets that we have used for testing our approach. Section 3.5 explains the complete experimental setup, including the wide range of evaluation metrics that we use for the validation of our generated paraphrases. Section 3.6 provides our insights and results from the experiments. Finally, we summarize the chapter in Section 3.7.

3.1 PROBLEM DEFINITION

A task-oriented dialogue agent understands user requests (an utterance) and performs required actions to satisfy user needs. E.g., a dialogue agent plays the song aspro mavro bill from youtube after hearing the user's request; Play the song aspro mavro bill from youtube.

U:	find	movies	playing	at	the	closest	movie	theatre
I:	SearchScreeningEvent							
S:	O	B-Movie-Type	O	O	O	B-Spatial-Rel	B-Loc-Type	I-Loc-Type

Figure 3.1: Example utterance with intent and slot labels from SNIPS [69]. The slot labels (*S*) are in BIO-format.

task-oriented dialogue agent

The two fundamental components in task-oriented dialog systems are intent classification and slot labeling, which produce a formal meaning representation for an utterance that the system can act upon to fulfill the user's request.

an utterance example

Figure 3.1 shows an example utterance from SNIPs dataset [32]. The utterance *find movies playing at the closest movie theatre* belongs to *SearchScreeningEvent intent* and the sequence of tokens is labeled with expressed slots. In our work, we refer to the combination of intent name & slot names, and slot values as an *interpretation* of the utterance. There has been significant performance on these tasks on benchmarking datasets like SNIPs [32], but this work focuses on the addition of a new feature.

reasons for data augmentation

A new feature is defined as a set of one or more intents and related slots that were not known to the system before. The main challenge with such an addition is the limited availability of seed training data. Since deep learning models are known for their data-hungry nature, it is difficult to train good intent & slot models for the new feature due to its limited training data. Manual data collection seems to be a plausible solution, but as explained in Chapter 1, it is an expensive and time-consuming process. Therefore, in this work, we aim to reduce the time and effort and propose an automatic way to augment seed training data. In the following sections, we aim to answer our motivating question:

Can we generate high-quality synthetic training examples with limited parallel data?

3.2 RELATED WORK

related work on data augmentation

Related work use machine translation [52] to address data collection problem if the feature exists for other languages. Do and Gaspers [37] use cross-lingual transfer learning to use such existing data. For an already active feature, Muralidharan et al. [129] use feedback signals from users like paraphrases or interruptions to obtain additional training data. Qiu et al. [139] leverage the paraphrase relationship between pairs of unlabeled & labeled utterances to deduce labels for the unlabeled utterances for the feature.

In contrast to related work, we do not assume labeled data availability in another language, user feedback, or unlabeled data when bootstrapping a new feature. Therefore, existing works by [26] and Malandrakis et al. [115] closely relate to our work, with the same setup of having limited seed training examples. Malandrakis et al. [115] generate paraphrases for seed data using conditional variational auto-encoders. However, they do not evaluate slot labeling and do not propose any technique to add slot labels to the generated paraphrases. Cho, Xie, and Campbell [26] use Transformer network for paraphrase generation and use baseline intent & slot model for self-labeling new utterances. Their experimental results show good performances on downstream tasks.

related work on data augmentation for limited seed training data

However, our approach is more data-efficient as we don't rely on any baseline model for self-labeling, where self-labeling itself employs training data. We leverage existing features and employ more advanced sampling strategies to obtain high-quality paraphrases from limited seed examples. Our experiments demonstrate the applicability of our approach to earlier stages of bootstrapping by using fewer seed data than Cho, Xie, and Campbell [26]. Furthermore, our paraphrase generation approach employs training an *interpretation-to-text* model instead of a *text-to-text* model, thus using single utterances with intents, slot names & slot values. Our approach is independent of any parallel data and uses existing data for the downstream tasks of intent classification and slot labeling.

our approach

Beyond our specific use-case, text-to-text models for paraphrase generation have gained a lot of attention in recent years [57, 99, 100, 137]. These models use paraphrasing datasets, like Quora Questions Pairs[1] and WikiAnswers[2], which mainly focus on questions. In our preliminary study, we observe the direct application of paraphrasing models by training in the dialog domain yields unnatural sentences, thus making their use inapplicable for scenarios like the one proposed in this work.

test-to-text paraphrase generation models

Our *interpretation-to-text* model is closely related to data-to-text approaches i.e. generating natural language from structured data. There are various versions of this task with slight variations in the input like a generation from abstract meaning representations [84], generation from tabular data [23], as well as unsupervised natural language generation using denoising autoencoders [48]. The E2E NLG challenge [45], a competition conducted on the restaurant domain E2E dataset [131], with slot-based input representations, holds a very strong resemblance to our paraphrase generation scenario.

analogy to data-to-text

1 https://www.kaggle.com/c/quora-question-pairs
2 http://knowitall.cs.washington.edu/paralex/

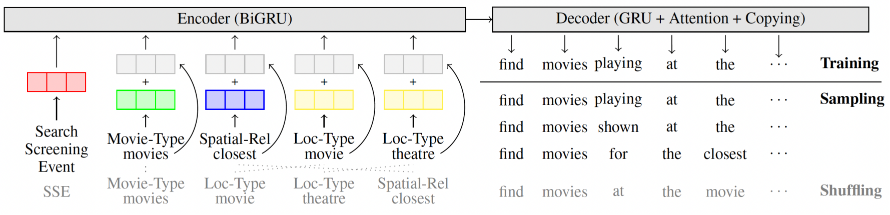

Encoder (BiGRU)
Decoder (GRU + Attention + Copying)
Search Screening Event
Movie-Type movies
Spatial-Rel closest
Loc-Type movie
Loc-Type theatre
SSE
Movie-Type movies
Loc-Type movie
Loc-Type theatre
Spatial-Rel closest
find movies playing at the ··· Training
find movies playing at the ··· Sampling
find movies shown at the ···
find movies for the closest ···
find movies at the movie ··· Shuffling

Figure 3.2: The interpretation-to-text model encodes an interpretation as a sequence of intent, slot name and slot value embeddings and is trained to predict the utterance. At inference time, we sample token by token and shuffle the input slot order (bottom) to obtain diverse paraphrases [69].

3.3 PROPOSED SOLUTION FOR DATA AUGMENTATION

formalization of new and existing features

We formalize $\mathcal{D}_N$ and $\mathcal{D}_O$ as the seed examples for the new feature and all the existing examples for the existing features. An example $(u, i, s) \in \mathcal{D}_N \cup \mathcal{D}_O$ comprises a tokenized utterance u, its true intent label i and true slot labels s in BIO-format. We refer to *slot name* as the label of a slot, and the covered tokens are referred to as *slot value*. We express the example in Figure 3.1 as $\hat{s} = \{$*Movie-Type* : *movies*, *Spatial-Rel* : *closest*, *Loc-Type* : *movie theatre*$\}$, where $\hat{s}$ is the mapping of slot names to their values.

formalization of $\mathcal{PG}$ model

In our scenario of adding a new feature, $\mathcal{D}_N$ is very small and contains one or a few new intents which are not present in $\mathcal{D}_O$. It is worth mentioning that $\mathcal{D}_N$ contains all new intents, however, slots might overlap with $\mathcal{D}_O$ as generic slots are shared across multiple intents. We propose $\mathcal{PG}$, as our paraphrase generation model, that generates new examples $\mathcal{D}_P = \mathcal{PG}(\mathcal{D}_N)$. Our $\mathcal{PG}$ aims to improve the performance of downstream tasks, intent classification, and slot labeling, when these models use all training examples $\mathcal{D}_O \cup \mathcal{D}_N \cup \mathcal{D}_P$ instead of just $\mathcal{D}_O \cup \mathcal{D}_N$.

In the following sections, we will describe our $\mathcal{PG}$ or *interpretation-to-text* model and the method of generating diverse paraphrases.

3.3.1 *Interpretation-to-text Paraphrase Generation Model*

We use examples (u, i, s) to model utterance sequences token by token conditioned on their corresponding interpretation. Equation 3.1 refers to our model,

$$p(u \mid i, \hat{s}) = \prod_{j=1}^{n} p(u_j \mid u_{1:j-1}, \mathrm{enc}_j(i, \hat{s})) \tag{3.1}$$

here, $\mathrm{enc}_j(i, \hat{s})$ refers to attention-based encoding of the interpretation $(i, \hat{s})$ at decoding step j. We motivate our approach by the notion of two utterances having the same interpretation are paraphrases.

Therefore, we model mapping from that unique representation of a set of paraphrases – the shared interpretation – to all its realizations. During inference, we use the conditional language model $p(u \mid i, \hat{s})$ which provides a distribution over all possible realizations conditioned on a specific interpretation, from which we sample the paraphrases.

We use the downstream training data $\mathcal{D}$ for training our $\mathcal{PG}$ model, thus making our approach data-efficient. Our approach allows us to include the large set $\mathcal{D}_O$ in addition to the seed data $\mathcal{D}_N$ for training a much more powerful $\mathcal{PG}$ model instead of using $\mathcal{D}_N$ only or using out-of-domain paraphrase data. This way, the model learns more general language properties specific to the dialogue system and similarities of utterances across multiple intents and slots. Furthermore, at the time of inference, the model can interpolate between utterances seen in $\mathcal{D}_N$ or $\mathcal{D}_O$ while sampling paraphrases for interpretations from $\mathcal{D}_N$ token by token. Therefore, it can create novel utterances, i.e., not seen during training for the interpretation.

interpretation-to-text model

Figure 3.2 shows our *interpretation-to-text* model. We use a sequence-to-sequence paradigm [169] to implement $p(u \mid i, \hat{s})$. We use a bidirectional GRU for encoding the interpretation and use an attention-based GRU along with a pointer mechanism for decoding [154]. We use GloVe vectors [136] to initialize embeddings for utterance tokens, intents, and slot names. The tokens whose embeddings are not present in GloVe are initialized randomly and updated during model training. As shown in Figure 3.2, the model takes an interpretation as the input. The sequence starts with the embedded intent followed by the sum of vector representations of slot names & values. We keep the vocabulary and embeddings the same for both encoder and decoder sides. The model is trained by minimizing cross-entropy loss for the utterances in $\mathcal{D}_O \cup \mathcal{D}_N$. Each of the encoder and decoder GRU has a size of 300 and contains two layers each. We use a batch size of 64, 0.3 dropout with early stopping.

3.3.2 *Sampling Strategies for Paraphrases*

For a given input, Seq2Seq models typically use greedy decoding or beam search to find the (approximately) best sequence at the inference times. In our use case, however, we are interested in obtaining novel & diverse utterances in addition to the most likely realization of an interpretation. Therefore, we use *random sampling* and *input shuffling*.

3.3.2.1 *Shuffling of Input Representation*

Our sequence-to-sequence model accepts input slots in a specific order; therefore, the simplest way to obtain multiple paraphrases can be obtained by shuffling this input order. We use shuffling of input rep-

shuffling input representation generates multiple paraphrases resentation and observe that the model using one order and decoding with an alternative order provides paraphrases with the values in an alternative order. Input shuffling allows us to obtain multiple paraphrases for the same interpretation, but the decoded sentences face the problem of missing a slot. This limitation motivates our quality metric *partial slot carry-over (PSCO)* represented by Equation 3.2;

$$\mathrm{PSCO}(u', \hat{s}) = \frac{1}{|\hat{s}|} \sum_{j=1}^{|\hat{s}|} \min(1, |\{t \,|\, t \in u' \wedge t \in \hat{s}_j^v\}|) \tag{3.2}$$

PSCO measures the fraction of slots $\hat{s}$ for which at least one token of the slot value $\hat{s}_j^v$ is present in the decoded utterance u'. We use *PSCO* to collect k paraphrases for a given seed utterance u. We start by creating input sequences corresponding to all permutations for the slot value. We compute the *PSCO* for each decoded utterance, which is decoded using beam search for each input. We keep utterances with a 100% *PSCO*, i.e., with a rate of 1. We sample k utterances from the remaining candidates if the candidates are more than k; otherwise, we upsample to reach k. Finally, we ensure that each seed leads to exactly k paraphrases and preserves the initial distribution.

psco based selection

3.3.3 *Non-deterministic decoding*

We use random sampling, a non-deterministic decoding strategy, as our second approach for generating novel and diverse utterances (paraphrases). Instead of trying to find the most likely sequence, as in beam search, random sampling samples an utterance token by token according to the probability distribution over the vocabulary. The non-deterministic nature of random sampling yields different utterances in different sampling rounds, thus allowing us to obtain multiple diverse utterances per seed.

multiple utterances/-paraphrases per seed using random sampling

$$p(u_j = t \,|\, u_{1:j-1}, h_j) = \frac{\exp(z_t/\alpha)}{\sum_{t' \in V} \exp(z_{t'}/\alpha)} \tag{3.3}$$

Equation 3.3 shows a unique technique to improve the outputs of any decoding algorithm. It scales logits z_t over the vocabulary with a temperature α before applying softmax while decoding the token u_j. After scaling, it then samples from the top-β tokens according to their probabilities. Here, a lower value of α makes the distribution spikier, thus encouraging the selection of more probable words. While higher α values make distributions uniform, i.e., allowing the selection of less probable words. The latter leads to more diverse & novel utterances but, at the same time, can produce unnatural utterances. α and β are hyperparameters and need to be tuned based on validation datasets.

scaling logits for generating diverse & novel utterances

3.3.4 *Projection of labels*

We need to turn the obtained paraphrases u' for a given seed example (u, i, s) into a useful example for the downstream tasks. For intent classification, we use the original label i. However, since u and u' are different utterances, it is intricate for projecting per-token slot labels. Nevertheless, considering a high overlap between tokens, we use a token alignment-based approach inspired by [50].

using intent i as intent label for paraphrase

We first compute a similarity $sim(u_j, u'_k) \in [0, 1]$ for each source-target token pair (u_j, u'_k). To identify identical tokens or slight morphological variations, we use the inverse of character-level Levenshtein edit distance normalized by length. We consider all alignments of source tokens in u to target tokens in u' based on their pairwise similarities and score them by their average similarity of chosen alignments. If no better target is found, the source is aligned to a virtual empty target token. We found the best alignment greedily and chose the most similar target for each source from left to right. At last, we restore prefixes and project slot labels s along that alignment. The final result s' forms the new training examples (u', i, s') for $\mathcal{D}_P$.

using alignment-based label projection to get token-level slot labels for paraphrase

3.4 DATASET

We use two datasets, a public dataset with English utterances and internal data from real-world dialog systems in German. We simulate introducing a new feature for both datasets to test our data augmentation approach.

SNIPs NLU Dataset [32] is a commonly used benchmark for the task of intent classification and slot labeling. There are seven intents and 39 slots. The dataset contains 13,084 training, 700 validation, and 700 test examples. We pick one of the intents as a new feature each time and run seven experiments. We use the full training and validation data of the remaining six intents as $\mathcal{D}_O^{train}$ and $\mathcal{D}_O^{val}$, amounts to 11,815 instances. We sample 5% of the data for our new feature as the seed data ($\mathcal{D}_N^{train}$, $\mathcal{D}_N^{val}$). The seed data amounts to 100 instances. We evaluate our approach on the standard test split of SNIPs. We perform several runs of our experiment and report the averaged results to overcome randomness due to the small size of the SNIPs dataset. We use each of the seven intents as a simulated new feature. We sample three different 5% seed subsets for the new feature. Finally, we train models for downstream tasks with ten different seeds, resulting in 210 different experimental runs per approach.

snips nlu dataset

experimental setup for snips

Our second dataset, in German, is randomly sampled data from logs of a **commercial dialogue system**. We simulate introducing five different features that have been introduced in the past. We use $|\mathcal{D}_N^{train}| =$ 450 and $|\mathcal{D}_N^{val}| = 50$ seed utterances for each of the simulated new feature. We have large datasets $\mathcal{D}_O^{train}$ and $\mathcal{D}_O^{val}$ that cover several hun-

real-life dialogue agent dataset

dred other intents. It is worth mentioning that $\mathcal{D}_O^{train}$ and $\mathcal{D}_O^{val}$ don't include any examples from any of the five simulated features. We use two disjoint test sets, one from the same distribution as $\mathcal{D}_O^{train}$, and the other containing new feature examples for the evaluation.

3.5 EXPERIMENTAL SETUP

3.5.1 *Downstream task models*

neural models for downstream tasks

We use a neural model for our downstream intent classification and slot labeling tasks. We use a bi-directional GRU with a hidden size of 512 to encode input tokens where the input tokens are embedded as 300-dimensional vectors. The concatenated final representations of GRU are fed through a 300-dimensional ReLU layer for intent classification. We apply dropout and a final softmax layer over the intent labels. We use a similar two-layer network for slot labeling, but instead of classifying the entire utterance, we perform classification at each timestep to predict slot labels per token. We use an Adam optimizer, a batch size of 64, and a dropout of 0.2 until performance convergence on the validation data. To study the effect of our data augmentation approach on both tasks, we trained these models separately.

3.5.2 *Approaches for Comparison*

We compare the following variations of our proposed techniques in Section 3.3.

variations of proposed technique

- **Baseline** Training the model using just the seed data $\mathcal{D}_N$ and existing data $\mathcal{D}_O$. All data augmentation approaches aim to perform better than this baseline model.
- **Upsampling5** Repeats each seed example five times to match the amount of training data for the new feature obtained using augmentation techniques below.
- **Beam1** Training model on $\mathcal{D}_O$, $\mathcal{D}_N$ and $\mathcal{D}_P$, where $\mathcal{D}_P$ is obtained from decoding, using beam search, one paraphrase for each seed in $\mathcal{D}_N$. We use a beam size of five and normalize scores by length.
- **Beam5** Same as Beam1, with the only difference of using the top-5 hypothesis from the beam for each seed.
- **Shuffle+Beam5** Adding input shuffling to Beam5. As described in Section 3.3.2.1, we select five paraphrases for each seed based on PSCO from the beams across all shuffles.

	Seed		Input / Output
A	i'm looking for a table at a pasta restaurant in serbia	1	*BookRestaurant, Dish=pasta, Type=restaurant, Country=serbia*
			i'm looking to find a table at a pasta restaurant in serbia
		2	*BookRestaurant, Dish=pasta, Country=serbia, Type=restaurant*
			book a pasta in serbia restaurant
		3	*BookRestaurant, Type=restaurant, Dish=pasta, Country=serbia*
			i'm looking for a restaurant that serves pasta in serbia
B	please play the newest music by evil jared hasselhoff	1	*PlayMusic, Sort=newest, Artist=evil jared hasselhoff*
			play the newest music by evil jared
		2	*PlayMusic, Artist=evil jared hasselhoff, Sort=newest*
			play music by evil jared hasselhoff on the newest
C	closest movie theatre with no time for sergeants	1	*SearchScreeningEvent, Sp.-Rel=closest, Movie=no time for ...*
			is the closest no time for sergeants at the movie theatre

Figure 3.3: Paraphrases of varying quality sampled for seeds from SNIPS using beam search and input shuffling (Shuffle+Beam5). Left side shows the seed, right the input and output of the paraphrase model. Note that the examples were manually chosen to demonstrate both good and bad examples [69].

- **Rand1** Same as Beam1, with the only difference in decoding strategy of random sampling. We sample with $\alpha = 2$ and $\beta = 3$[3].
- **Rand5** Same as Rand1, and we are sampling five paraphrases for each seed.
- **Shuffle+Rand5** Same as Shuffle+Beam5, with the difference in decoding strategy (we use random sampling). We sample three paraphrases for each shuffle and then select five across all shuffles using PSCO (Section 3.3.2.1).

Our data augmentation approach added 500 & 2500 examples for 100 & 500 seed examples for SNIPs and commercial dialogue system data. It would be interesting to explore more seed-paraphrase-ratios for determining an optimal one. Still, we leave this for future work and focus mainly on comparing different sampling techniques in this work.

3.5.3 *Evaluation Metrics*

Since we aim to boost performance for downstream tasks, our primary evaluation metric is the performance of these tasks. We use accuracy for intent classification and slot F1-score for slot labeling. We compare all our experiments with the baseline to see the impact of data augmentation for bootstrapping a new feature. We provide separate scores for the new feature and the existing features. It allows

3 We manually inspected paraphrases for a range of parameter settings and found these parameters to represent a good trade-off between novelty and quality.

	Upsampling5	Beam1	Beam5	Shuffle +Beam5	Rand1	Rand5	Shuffle +Rand5
PSCO	1.000	0.988	0.984	**1.000**	0.911	0.910	**1.000**
ESCO	1.000	0.969	0.956	**0.972**	0.773	0.772	0.878
Novelty	0.000	0.486	0.574	0.742	0.767	0.766	**0.864**
Diversity	0.000	–	0.522	0.594	–	0.837	**0.881**

Table 3.1: Quality comparison of paraphrases generated on SNIPS with different approaches along our four metrics. Bold marks best technique per metric (excluding the edge case Upsampling5) [69].

us to see if bootstrapping a new feature is detrimental to the existing ones or not.

We adopt some additional metrics to gain further insights into the relative strengths of different paraphrase generation approaches.

evaluation metrics

- **Exact Slot Carry Over (ESCO)**, a stricter version than PSCO, counts only slots whose complete slot value is present in the generated paraphrase.
- **Novelty**, We compute Novelty scores using BLEU [133]. For our use case, we want the new example to be different from the original seed example; otherwise, the data augmentation degenerates to simply upsampling the data. Therefore, we compute the score $1 - \text{BLEU4}(u, u')$. for each paraphrase u' of seed s and aim to achieve higher scores for this metric. We report average over all sampled paraphrases.
- **Diversity**, The score checks if the sampled paraphrases are different or it's the repeated sampling of the same paraphrase. We compute $1 - \text{BLEU4}(u', u'')$, where u', u'' are paraphrases of the same seed, to measure diversity. We report the average over all pairs and aim to achieve higher diversity scores.

All these metrics guide a better understanding of paraphrases sampled with different methods. In our results, we will show that higher novelty or diversity is beneficial for the performance of downstream tasks.

3.6 RESULTS

3.6.1 *Qualitative Analysis of Generated Paraphrases*

qualitative analysis

We have presented several paraphrase examples in Figure 3.3, which are obtained by our approach. The generated utterances differ from the original seed example by dropping some tokens (B1, A1). The change in input order leads to paraphrases with more dropping of

SNIPS	New Feature					Existing Features			
Method	#Train	IC	Δ	SL	Δ	IC	Δ	SL	Δ
Baseline	100	88.1		52.1		98.9		88.8	
Upsampling5	600	90.5	+2.46	63.6	+11.47	98.8	-0.10	88.4	-0.40
Beam1	200	89.7	+1.67	57.7	+5.58	98.8	-0.06	88.7	**-0.08**
Beam5	600	90.6	+2.50	62.9	+10.78	98.8	-0.10	88.5	-0.23
Shuffle+Beam5	600	88.7	+0.67	63.1	+11.02	98.7	-0.15	88.5	-0.23
Rand1	200	91.0	+2.97	57.5	+5.36	98.9	**-0.02**	88.6	-0.11
Rand5	600	92.0	**+3.95**	63.4	+11.32	98.8	-0.14	88.5	-0.27
Shuffle+Rand5	600	91.3	+3.26	64.7	**+12.66**	98.8	-0.14	88.4	-0.31

Table 3.2: Intent classification (IC) accuracy and slot labeling (SL) F1-scores on the SNIPS test set (English) after adding generated paraphrases as additional training data. Δ denotes the absolute change with regard to the baseline. Each result is an average over 210 runs of the experiments [69].

words (A2, A3), which in the extreme can also become unnatural and ungrammatical (B2, C1).

Table 3.1 shows comparison between generated paraphrases from different approaches as defined in Section 3.5.2. It clearly shows that the decoding strategy of random sampling generates novel & diverse paraphrases than from beam search. However, this diversity comes at the expense of carrying over fewer slots, which indicates that too novel paraphrases might not fully represent the intended interpretation. As mentioned in Section 3.3.2.1, we keep a PSCO of 1, but the ESCO shows a lower carry-over rate for random sampling. It is worth noting that this tradeoff between slot carry over and novelty/-diversity is balanced with input shuffling, where input shuffling leads to the highest slot carry over along with higher novelty & diversity scores for both decoding strategies. Hence, our proposed combination of shuffling and PSCO-based selection proves to be an effective way that improves paraphrases across all dimensions.

input shuffling+random sampling generates more diverse & novel paraphrases

3.6.2 *Downstream Task Performance*

Table 3.2 presents the performance of downstream tasks, on SNIPs, after adding paraphrases generated with different approaches. The generated paraphrases (500 vs. 100) using both sampling strategies improve downstream tasks of intent classification & slot labeling. In line with the increased novelty and diversity scores from Table 3.1 for random sampling, the decoding strategy helps more than the beam search. However, due to the coupling of lower slot carry over with higher diversity & novelty, the gains for slot labeling are less pronounced. The shuffling of input representation shows higher gains

diverse & novel samples help downstream tasks

Internal Data	New Feature				Existing Features			
	ICΔ		SLΔ		ICΔ		SLΔ	
New Feature	S+B5	S+R5	S+B5	S+R5	S+B5	S+R5	S+B5	S+R5
WeatherForecast	+1.48	+4.81	+3.91	+4.98	-0.02	-0.04	-0.02	-0.01
SendMessage	+4.84	+7.73	+0.65	+0.47	-0.13	-0.11	-0.01	-0.02
PlayMusic	+8.79	+10.12	+1.35	-0.12	+0.11	+0.10	-0.05	-0.13
MovieListing	+9.50	+10.32	+1.28	+0.92	+0.05	+0.05	-0.09	-0.06
ApplianceOnOff	+0.91	+12.53	+1.25	+0.79	-0.02	-0.18	+0.04	-0.07
Average	+5.10	+9.10	+1.69	+1.41	0.00	-0.04	-0.03	-0.06

Table 3.3: Intent classification (IC) and slot labeling (SL) performance change on internal data (German) across 5 (simulated) new features when adding paraphrases generated with Shuffle+Beam5 (S+B5) or Shuffle+Rand5 (S+R5). Numbers are absolute changes with regard to the baseline [69].

for slot labeling but minimal gains for intent classification, which could be because of less relevance of word order for this task. It is worth noting that compared to the improvements on the new feature, the performance drop for existing features is negligibly small. At last, Upsampling5 seems to be a strong baseline, beating several of our ablations, but our complete method outperforms it. It indicates that the diverse and novel paraphrases are beneficial for the downstream tasks of intent classification and slot labeling.

our approach helping real-life German dialogue system

We only compared our complete method, which includes input shuffling, across two sampling methods on the commercial data. Table 3.3 shows that the average improvement over all simulated new features achieved is even bigger for intent classification. Similar to SNIPs, random sampling performs better than beam search. Also, as desired, the existing features face negligible drops compared to the improvements on the new feature. Furthermore, in contrast to SNIPs, the commercial data shows lower improvements for slots using random sampling. We attribute these small improvements to the fact that the model already knows many (shared) slots due to larger sizes of existing features, thus making it hard to improve the baseline, which is already performing better. Table 3.3 shows breakdown by the new feature. Due to the degrees of variety in utterances for specific features and their similarity to existing features, changes are different across features.

3.7 SUMMARY

We started this chapter to answer our motivating question:

Can we generate high-quality synthetic training examples with limited parallel data?

We introduced a scenario of adding new functionality to a dialogue agent, where limited training data is available for this new functionality. We proposed a **data augmentation** approach to generate synthetic data such that adding this data improves performance on downstream tasks of intent classification and slot labeling. Our approach is independent of any parallel data and leverages already available training examples. We designed an *interpretation-to-text* model that generates paraphrases, which we use to augment our training data for the newly added feature. We presented the role of different decoding strategies in generating diverse & novel paraphrases. The impact of shuffling the input representations in generating more utterances was also shown. We also presented an alignment-based label projection to obtain token-level slot labels. We tested our approach on two datasets: SNIPs, an open-source dataset in English, and a commercial dialog system dataset in German. We presented multiple evaluation metrics as *exact slot carry over*, *diversity*, and *novelty* scores to measure the quality of generated paraphrases and *partial slot carry over* for selecting the best paraphrases for data augmentation. We designed neural models for downstream tasks. Multiple approaches for comparisons were also presented. It was shown that diverse and novel utterances are helpful for downstream tasks. The proposed method of generating data improves performance for intents & slots on an English benchmark and German dialog system data.

In this chapter, we studied the scenario of the availability of limited labeled examples from one intent. However, the availability of training examples from all other intents enabled knowledge sharing, leading to high-quality text generations. But what if we have limited training data from all the intents, i.e., a few-shot training setting. A few-shot setting will not allow knowledge sharing and could lead to performance degradation on the entire test set. We will delve into this scenario and leverage another method called distant supervision (Section 2.4) to overcome accuracy drops arising from limited training examples.

4 IMPROVING DISTANT SUPERVISION USING UNSUPERVISED NLG ALGORITHM

As discussed in Chapter 3, we will study a few-shot scenario in this chapter, i.e., we have minimal training examples. Similar to the task in the last chapter, we discuss tabular data summarization or data-to-text generation. It involves generation of textual descriptions for a given tabular data (Section 2.2.1). We observe that tabular data summarization faces a *low semantic coverage* problem in a few-shot setting where important input table values (slots) are missing in the generated sentence. To overcome this problem, we use a small unlabeled dataset and employ distant supervision (Section 2.4) to generate sentence outputs for these unlabeled tables to augment limited training examples. However, the *low semantic coverage* problem persists. Therefore, we propose a search-and-learn algorithm that exploits pre-trained language models (e.g., T5 [143]) to fill the missing slots and improve semantic coverage. We use the search results to finetune our system to smooth out search noise, producing fluent text and improving inference efficiency significantly. We test our algorithm on two open-source data-to-text datasets. Our method recovers a majority of missing input table slots on these datasets, largely alleviating the *low coverage problem* and closing the gap between few-shot and fully supervised learning.

publication and chapter structure

The proposed algorithm and results presented in this chapter have been published in AAAI 2022 [74]. We have organized the rest of the chapter as follows. Section 4.1 explains the problem definition. We discuss related work in Section 4.2. Section 4.3 discusses the problem formalization. We present our proposed unsupervised algorithm in Section 4.4. We discuss the experiments in Section 4.5. The section comprises two experiments on two datasets. We explain implementation details and result analysis for both datasets. We also discuss a detailed analysis of the E2E dataset in Section 4.6. Finally, we conclude this chapter with a summary in Section 4.7.

4.1 PROBLEM DEFINITION

applications for data-to-text

Data-to-text converts structured data into human-readable text descriptions. The data-to-text generation has gained significant attention in the field of natural language processing, with its applications in multiple fields like restaurants [131], weather forecasts [101], and biographies [95]. Before the introduction of neural-based models, some traditional approaches used handcrafted rules with statistics

[92, 149, 166] for text generation. However, the generated text lacks flexibility and output diversity, thus motivating the use of modern neural networks like Seq2Seq recurrent neural networks [95, 106].

dependency of neural-based models on large training datasets

These Seq2Seq models use massive parallel training datasets to generate fluent human-readable sentences. For eg., one of the crowd-sourced data-to-text datasets, namely E2E [131], contains 42k table-text training pairs. As explained in Chapter 1, these huge training examples are expensive due to the amount and effort involved in data collection and hence make data-to-text an expensive and time-consuming affair and restrict its real-world applications.

Therefore, in this work, we focus on a few-shot setting, i.e., considering a scenario where we have very few parallel training examples. We use pre-trained language models and fine-tune them using our few-shot training examples.

problems in a few-shot setting

In contrast to traditional neural networks, these pre-trained language models leverage massive amounts of unlabeled corpora to learn generic knowledge of the natural language for generating fluent text using fewer examples. However, we observe that, in a few-shot setting, these fine-tuned language models miss information slots in the generated text and thus face the problem of *low-semantic coverage*. We handle this problem by proposing a search-and-learn algorithm that achieves high semantic coverage of input slots. In the following sections, we aim to answer our motivation question:

Can we minimize semantic information loss when training NLG systems with minimal training data?

4.2 RELATED WORK

data-to-text generation

The generation of human-readable textual descriptions from tabular data has been a persistent problem from early NLP research. The traditional works follow a pipeline approach of content planning and surface realization [146]. There have been works using hand-engineered rules [88, 121] and the ones that employ statistical induction [83, 101]. Despite generating text, these approaches usually lack flexibility and diversity. The recent success of neural-based models for data-to-text generation [95, 105, 106, 156, 187], handles this problem by learning from massive parallel data for generating fluent and human-readable tabular descriptions.

few-shot data-to-text generation

Chen et al. [24] is one of the recent works in few-shot learning to data-to-text generation. They finetune pre-trained language models (LMs) with a copy mechanism. They assume the availability of a small parallel corpus. We observe that despite generating fluent sentences, these LMs face the problem of *low-semantic coverage*. It is, therefore, difficult to "force" a copy mechanism to copy the entire source information.

The problem of low semantic coverage has been observed in the human-written references, which motivated the introduction of the new metric for better evaluation of data-to-text models by Dhingra et al. [36]. Another work by Gong et al. [54] focuses on the fidelity of data-to-text generators. Along with the finetuning of GPT, they use table reconstruction and content matching. In our preliminary analysis during development, we observe that T5 does not generate wrong information but may miss input slots.

Researchers have presented various search approaches to address unsupervised text generation, like simulated annealing [107] and hill-climbing [89, 153]. Their approach involves a heuristic objective function typically involving semantic coherence, language fluency, and other task-specific scores and text generation by word-level editing towards the objective. These approaches, however, led to a high inference complexity. Li et al. [97] propose a search-and-learning approach that both improves model performance and inference complexity.

unsupervised text generation using search approaches

Similar to Li et al. [97], we also use a search-and-learning framework with the following differences from the previous approaches: 1) We address the few-shot setting. We use a fine-tuned language model to evaluate candidate sentences instead of a heuristically defined objective. 2) Our search aims for higher semantic coverage rather than a generic fluent sentence. The main focus of our work is not the search algorithm, as the search space is relatively simpler than the entire sentence space. We adopt a greedy search over multiple missing slots and are the first to address few-shot data-to-text generation by search and learning.

our contributions

There have been some works that learn word edits in a supervised way [40] or treat rule-edited text as input [98, 183]. However, different from them, we perform editing as search steps and learn from the editing results.

4.3 FORMALIZATION OF PROBLEM

The data-to-text generation generates natural language textual descriptions for structured input data, which is in tabular form in our scenario. Each input table is a set of slot name–value pairs, denoted by $\mathbf{T} = \{(s_i, v_i)\}_{i=1}^{S}$, where s_i is the name of the ith slot, v_i is the value, and S represents the number of slots. $\mathbf{y} = (y_1, y_2, \cdots, y_n)$ is the output sentence that describes the given input $\mathbf{T}$. Here, each example has a different input table $\mathbf{T}$, but for clarity, we may omit the example index.

formalization of few-shot setting

In our few-shot setting, we have a small parallel corpus denoted as $\mathcal{D}_p = \{(\mathbf{T}^{(i)}, \mathbf{y}^{(i)})\}_{i=1}^{M}$ and another small unlabeled corpus $\mathcal{D}_u =$

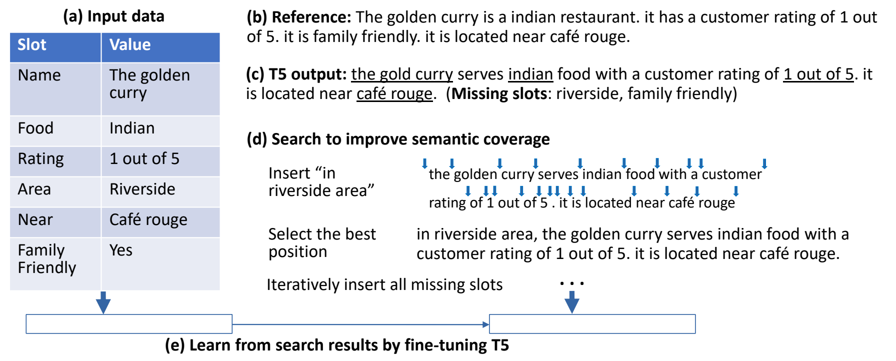

(a) Input data
Slot
Value
Name
The golden curry
Food
Indian
Rating
1 out of 5
Area
Riverside
Near
Café rouge
Family Friendly
Yes
(b) Reference: The golden curry is a indian restaurant. it has a customer rating of 1 out of 5. it is family friendly. it is located near café rouge.
(c) T5 output: the gold curry serves indian food with a customer rating of 1 out of 5. it is located near café rouge. (Missing slots: riverside, family friendly)
(d) Search to improve semantic coverage
Insert "in riverside area"
the golden curry serves indian food with a customer rating of 1 out of 5 . it is located near café rouge
Select the best position
in riverside area, the golden curry serves indian food with a customer rating of 1 out of 5. it is located near café rouge.
Iteratively insert all missing slots
...
(e) Learn from search results by fine-tuning T5

Figure 4.1: An example for data-to-text generation and our proposed approach [74].

$\{\mathbf{T}_u^{(i)}\}_{i=1}^N$. Here, $\mathbf{T}_u^{(i)}$ and $\mathbf{T}^{(i)}$ are different tables and both M and N are small numbers in our few-shot setting.

In Natural Language Generation, few-shot learning plays a significant role as it saves human annotation labor. It also alleviates the cold-start problem of new NLG tasks. In our work, we assume the availability of an unlabeled corpus $\mathcal{D}_u$ in addition to $\mathcal{D}_p$, where the sizes of both $\mathcal{D}_u$ and $\mathcal{D}_p$ are small. Since the collection of unlabeled data is easier than the labeled pairs containing human-written sentences, our proposed scenario resembles a realistic setting for few-shot learning. Furthermore, as discussed in Section 3.3.2.1,$\mathcal{D}_u$ can also be synthesized by recombining the slots of $\mathcal{D}_p$ for a data-to-text generation.

4.4 PROPOSED SOLUTION

Our approach comprises three steps. First-Stage Fine-tuning T5; we first finetune a pre-trained language model (LM) for conditional text generation based on input tables. Due to extensive pre-training, a language model can generate fluent sentences in a few-shot setting. However, these models fail to fully learn the correspondence between input table slots and generated text. It faces the problem of *low-semantic coverage*. Therefore, we perform a Search to Improve Semantic Coverage, iteratively adding a missing slot into the generated sentence in a greedy manner. Finally, we perform Second-Stage Fine-Tuning T5 with Search Results, where we treat the search results as pseudo-ground-truth for further finetuning of our language model. This final step improves both the inference efficiency and generates fluent sentences. We will explain each of these steps in the following sections.

overview

4.4.1 *First-Stage Fine-tuning T5*

We use the T5 model [143] for first-stage finetuning for a data-to-text generation. As explained in Section 2.1.3 of Chapter 2, T5 is a text-to-

If	Then the phrase template is
SN = food	SV food
SN = pricerange; SV is a number	price range SV
SN = pricerange; SV is a string	SV price range
SN = eattype	SV
SN = name	SV
SN = near	near SV
SN = family friendly; SV is yes	family friendly
SN = family friendly; SV is no/not	not family friendly
SN = customer rating	SV customer rating
SN = area	in SV area

Table 4.1: Rules for the E2E dataset.

text Transformer [173] model, pre-trained on multiple NLP tasks, and have achieved state-of-the-art performance in sentiment classification, question answering, document summarization etc. We must note that none of the pre-training tasks relate to data-to-text generation. Therefore, our experiments are in the few-shot setting, even if we use a pre-trained language model T5.

We linearize the input table by concatenation of all table slots as "`name[value]`". Here, "`[`" is a special token that separates the slot name and slot value. Another special token "`]`" separates different slots.

leverage pre-trained model to generate fluent text

In this step, we finetune T5 using a small corpus of a few hundred data-text pairs. This corpus is considerably smaller than the usual NLG training sets. Equation 4.1 presents the method of learning the conditional probability $P(\mathbf{y}|\mathbf{T})$ in an autoregressive way.

$$P(\mathbf{y}|\mathbf{T};\theta) = \prod_{i=1}^{n} P(y_i|\mathbf{y}_{<i},\mathbf{T};\theta), \tag{4.1}$$

Here, $\mathbf{T}$ is the input table, $\mathbf{y}$ is the output with length n, and θ represents the model parameters. Equation 4.2 presents the equation to finetune T5 with the cross-entropy loss.

$$J(\theta) = -\log P(\mathbf{y}|\mathbf{T};\theta) \tag{4.2}$$

4.4.2 *Search to improve Semantic Coverage*

In our experiments, we observe that T5 certainly generates fluent sentences, but it faces the problem of low semantic coverage. Figure 4.1 clearly shows this problem in which two slots, namely "riverside" and "family-friendly", are absent in the generated text. This coverage problem arises from the limited training data as the model

If	Then the phrase template is
SN = fullname	SV
SN = birth date	born on SV
SN = currentclub	plays for SV
SN = nationality	SV
SN = position	SV
SN = occupation	is a SV
SN = death rate	died on SV
SN = party	serving in SV party
SN = birth place	born in SV

Table 4.2: Rules for the WikiBio dataset.

fails to learn the correspondence between input and output from limited data–text pairs. Our experiment of analyzing the coverage percentage and the training size provides supporting evidence. The T5 model fine-tuned with 1% of E2E data has coverage of 84.46% input slots, whereas fine-tuning using entire training data gives coverage of 97.74%.

We start by first checking if each slot value v_i is present in the T5's output. This is possible either using the verbatim match of the slot value or a soft match based on a script by Dušek, Howcroft, and Rieser [44] that finds missing slots using regular expressions. We insert a phrase $\tilde{v}_i$ if the slot value v_i is absent in the output sentence. We design phrase $\tilde{v}_i$ such that it contains some supporting prepositions along with the original slot value. For example, we insert the phrase $\tilde{v}_i =$ "`in riverside area`" if slot "`area[riverside]`" is not present in the output. For Boolean slots, like "`familyFriendly[yes/no]`" in the E2E dataset, we design phrase as either "`not family friendly`" or "`family friendly`". One must note that the design of these phrases does not require much human labor as we have atmost ten phrases for each dataset. Also, most of these phrases involve copying the slot value. Table 4.1 and Table 4.2 presents the complete list of our phrases for both E2E and WikiBio datasets.

project infeasible solution to feasible set

We determine the most appropriate position to insert the missing slot. We select the candidate with the highest T5 probability $P(\mathbf{y}|\mathbf{T})$, as finetuned by Equation 4.1, shown in Figure 4.1, after the enumeration of all possible positions within a sentence. We greedily repeat this process for all the missing slots that we want to insert. We insert all the slots for the E2E dataset. However, for WikiBio, we select the desired slots by statistics as the task doesn't require the presence of all slots in the output sentence. We can consider our approach optimization towards

$$\text{maximize} \quad P(\mathbf{y}|\mathbf{T}), \quad \text{subject to } v_i \in \mathbf{y}, \forall i \tag{4.3}$$

Algorithm 1 Search and Learn [74]

```
Input: Small parallel data $\mathcal{D}_p = \{(\mathbf{T}^{(m)}, \mathbf{y}^{(m)})\}_{m=1}^{M}$
Small unlabeled data $\mathcal{D}_u = \{\mathbf{T}^{(n)})\}_{n=1}^{N}$
Pre-trained language model T5
Output: Few-shot learned data-to-text model
▷ First-stage fine-tuning T5
for $(\mathbf{T}, \mathbf{y}) \in \mathcal{D}_p$ in each epoch do
    Fine-tune T5 by minimizing $-\log P(\mathbf{y}|\mathbf{T})$
▷ Search to improve semantic coverage
$\widetilde{\mathcal{D}}_p = \emptyset$
for $\mathbf{T}_u \in \mathcal{D}_u$ do
    $\hat{\mathbf{y}}_{\text{search}} = T5(\mathbf{T}_u)$    ▷ search to be performed
    for missing slot $(s, v) \in \mathbf{T}_u$ that $v \notin \hat{\mathbf{y}}_{search}$ do
        Update $\hat{\mathbf{y}}_{\text{search}}$ by inserting $v$ with templates into the most appropriate position
    $\widetilde{\mathcal{D}}_p = \widetilde{\mathcal{D}}_p \cup \{(\mathbf{T}_u, \hat{\mathbf{y}}_{\text{search}})\}$
▷ Second-stage fine-tuning T5
for $(\mathbf{T}, \mathbf{y}) \in \mathcal{D}_p \cup \widetilde{\mathcal{D}}_p$ in each epoch do
    Fine-tune T5 by minimizing $-\log P(\mathbf{y}|\mathbf{T})$
Return: Second-stage fine-tuned T5
```

In other words, we start from an infeasible solution, an output violating the constraint, and project the solution into the feasible set by greedily satisfying each constraint.

The recent development of search-based unsupervised text generation, such as simulated annealing for paraphrasing [107] and hill-climbing for summarization [153], inspires our search approach. However, in contrast to these approaches that search for a generic sentence maximizing a heuristically defined objective, our search is devoted to projecting an infeasible solution to the feasible set. Furthermore, our approach differs significantly from template-based text generation systems [92, 166], as we use rules only for revision rather than defining rules for output generation.

using search for revision, not generation

The generated sentences at this step are often inflexible and disfluent text. Therefore, we design a learning component that learns from these search results to smooth out disfluent text.

4.4.3 *Second-Stage Fine-tuning T5*

As discussed in Section 4.4.2, the search approach aims to achieve complete coverage of input slots in the generated text. However, it has significant drawbacks like the disfluent generated text due to fixed templates and low inference efficiency when evaluating multiple candidate outputs for a given data example. Therefore, we perform second-stage finetuning of T5, inspired by Li et al. [97] to over-

search generates disfluent sentences with high semantic coverage

search increases inference complexity

come these drawbacks. This way, we learn from search results obtained from the last step. As discussed in Section 4.3, we assume the availability of a small unlabeled corpus $\mathcal{D}_u$ that only contains input tables. In practice, $\mathcal{D}_u$ is inexpensive to obtain and can even be synthesized by recombining the table slots and values in $\mathcal{D}_p$.

search-and-learn generates fluent text with high semantic coverage

We use T5 to generate candidate output for a given input table $\mathbf{T}_u^{(i)} \in \mathcal{D}_u$ and perform a search to ensure that the candidate output has high semantic coverage. We treat this search result as pseudo-groundtruth and denote it as $\hat{\mathbf{y}}_{\text{search}}^{(i)}$. This way, we create a pseudo-parallel corpus $\widetilde{\mathcal{D}}_p = \{(\mathbf{T}_u^{(i)}, \hat{\mathbf{y}}_{\text{search}}^{(i)}) : \mathbf{T}_u^{(i)} \in \mathcal{D}_u\}$. We mix this pseudo-parallel corpus with our original parallel corpus and use this combined dataset $(\mathcal{D}_p \cup \widetilde{\mathcal{D}}_p)$ to finetune T5. We use the same cross-entropy loss as Equation 4.2 to finetune T5. We summarize over training algorithm in Algorithm 1.

use second-stage finetune T5 for inference

We use the second-stage finetune T5 for **inference** on the test set. The second-stage T5 model is finetuned with the search results. We achieve better inference efficiency than the search approach as we do not perform the search during inference. Our approach leverages the power of pre-trained language models and generates more fluent sentences than the search itself. In contrast to the first-stage finetuned model, T5 explicitly learns from the pseudo-ground truth with high semantic coverage. Our experimental results show the effectiveness of our approach in which S&L achieves near-perfect semantic coverage on the E2E dataset.

4.5 EXPERIMENTS

4.5.1 *Experiment I: E2E Dataset*

Dataset We use E2E[1] [131], a crowd-sourced dataset for data-to-text generation, for evaluating our approach. It is from the restaurant domain. It contains more than 50K table–text pairs, and the input for each data example contains 3–8 slots. We follow the standard train/val/test split.

Implementation Details We use the T5-small model. There are six layers in the encoder and the decoder. We trained the model using the AdamW optimizer [110]. We use a learning rate of 3e-4 and pass 64 examples in one batch.

Evaluation For E2E, we used the standard evaluation scripts that are released along with the E2E dataset by [131]. The script computes BLEU [133], NIST [38], METEOR [94], CIDEr [174], and ROUGE-L [102] scores. We have explained each of these metrics in Section 2.2.4.1 of Chapter 2.

1 http://www.macs.hw.ac.uk/InteractionLab/E2E/

Recently, Dhingra et al. [36] proposed a set of PARENT (including precision, recall, and the F-score) metrics against both the input data and ground truth reference. They observe that BLEU scores do not correlate well with human satisfaction for the data-to-text generation, and PARENT scores highly correlate with human judgment. Considering this evidence, we consider PARENT as the principal metric. We have used the word-overlap version PARENT-W in our work.

Furthermore, we estimate the fluency of the generated text using GPT-2 perplexity (without fine-tuning). We have also computed average sentence length (AvgLen) for reference. We also calculated the fraction of input slots that appear verbatim in the output, called semantic coverage ratio or Hard Coverage. Although hard coverage seems strict, it gives a good approximation because most slots contain only one or a few words, and some are proper nouns that should not change.

We also computed slot error rate [SER, 44] which is specifically designed for the E2E dataset. Based on manually created regular expressions, SER checks if all E2E slot values are present, absent, incorrect, or are missing in the generated text, as represented by Equation 4.4:

$$\text{SER} = \frac{\#\text{added} + \#\text{missing} + \#\text{wrong value}}{\#\text{slots}} \tag{4.4}$$

here *#added* denotes the number of additional slots that were not in the table, *#missing* denotes the number of missing slots that were originally present in the table, *#wrongvalue* denotes the number of slots with incorrect slot values, and *#slots* denotes the number of total slots.

Since SER accounts for soft matching, we have used $1 - \text{SER}$ as a measure for Soft Coverage. At last, we have conducted a human evaluation on a randomly selected subset of test examples for E2E. The human evaluation results for the coverage percentage (Table 4.5) are close to the automatic metrics.

Results In our few-shot setting, we consider 1% of parallel examples ($\mathcal{D}_p$) and other 4% examples ($\mathcal{D}_u$) with only input tables. As shown in Table 4.3, we have two "upper bound" performances obtained by fine-tuning T5 with 100% examples (Line 4) and 5% examples (Line 5). We have shown results of the previous state-of-the-art models like TGEN [131], SLUG [75], and the S_1^R model [159] to motivate our setup where T5 (Line 4) achieves similar scores to these models.

We started few-shot learning by fine-tuning T5 on a small parallel training data. This fine-tuning led to poor performances in all metrics, as shown in Lines 4–6 of Table 4.3. More importantly, we observe a significant reduction in the hard and soft coverage scores, dropping quickly from more than 97% to around 84.19%.

slot coverage drops in few-shot setting

											Coverage (%)		
#	Model	#Train	BLEU	NIST	METEOR	RougeL	CIDEr	PARENT (P/R/F1)	PPL	AvgLen	Hard	SER	Soft
1	TGEN	p:42K	65.93	8.61	44.83	68.50	2.23	–	–	–	–	4.27	95.73
2	SLUG	p:42K	66.19	8.61	44.54	67.72	–	–	–	–	–	–	–
3	S_1^R	p:42K	**68.60**	8.73	45.25	70.82	2.37	–	–	–	–	–	–
4	T5	p:42K	67.59	**8.81**	**45.17**	**70.44**	**2.33**	67.40/61.75/63.43	154.49	23.58	97.50	2.62	97.38
5	T5	p:2100	62.45	8.30	44.10	67.15	2.17	64.25/61.69/62.00	136.54	24.82	96.21	3.72	96.28
6	T5	p:420	**61.72**	7.96	40.52	65.61	1.96	65.63/57.25/60.10	**141.61**	21.97	84.19	16.68	84.32
7	T5 self-train	p:420,u:1680	60.83	7.74	39.85	66.36	1.95	66.60/57.31/60.63	154.74	21.50	81.82	17.97	82.03
8	T5 S&L	p:420,u:1680	60.70	8.13	43.60	65.84	**2.12**	**66.97/63.63/64.29**	160.40	25.01	98.35	1.84	98.16
9	T5 S&L w/ SER	p:420,u:1680	60.89	**8.14**	**43.71**	**66.76**	2.07	65.16/62.97/63.04	170.26	25.71	**99.46**	**0.80**	**99.20**

Table 4.3: Test results on E2E. "p:" and "u:" denote the number of parallel and unlabeled training examples, respectively. Baseline results are quoted from original papers. All results of fine-tuning T5 are obtained by our experiments. Based on the evidence in [36], we consider PARENT as our main metrics. Hard Coverage measures the verbatim coverage of slot values presented in text. The slot error rate measures the percentage of missing, added, or wrong slots in the generated sentence. We consider Soft Coverage as $1 - \text{SER}$.

To handle this drop, we investigate if we can use the small unlabeled dataset $\mathcal{D}_u$ to boost this performance. We experimented with the standard strategy for semi-supervised machine learning known as self-training [200], a method for distant supervision. In this competing method, we first fine-tune T5 using the small parallel corpus $\mathcal{D}_p$. Furthermore, we use the trained T5 model to predict the outputs for unlabeled tables in $\mathcal{D}_u$. We treat these inferred sentences as pseudo-ground-truth and use them for further fine-tuning. Line 7 shows the result of this approach, and we find that the strategy doesn't help the performance (Lines 6–7).

no improvements from distant supervision

Line 8 and Line 9 of Table 4.3 presents two variants of our S&L approach. The first variant (Line 8) uses a verbatim match to determine missing slots, and the second variant (Line 9) uses SER to determine missing slots. Both these variants achieve higher performance than other few-shot models in most metrics. Specifically, our model achieves 98–99% coverage of input slots, thus solving the problem of low semantic coverage.

s&l solving low semantic coverage

Table 4.3 shows high perplexity (PPL) scores which indicates that the model generates less fluent sentences[2]. However, we notice that our sentences are longer and contain more input slots. These slots are often precise information like restaurant name as a proper noun for the E2E dataset, thus explaining the higher values of PPL. In general, all the models lie in the same ballpark of fluency. Our human evaluation study further analyzes fluency.

In comparison to T5 trained with four times more parallel data (Line 5), our approach achieves comparable results (Lines 8 and 9) in several metrics, such as METEOR and CIDEr. We observe that our approach outperforms Line 5 with a reasonable margin for PARENT metrics, the ones specifically designed for the data-to-text generation. It is worth noting that our approach achieves close PARENT and coverage scores to the fully supervised setting (Line 4).

s&l achieve highest parent & coverage scores

4.5.2 *Experiment II: WikiBio Dataset*

Dataset We use the humans domain of WikiBio dataset[3] [95] to further evaluate our approach. There are 700K English biographies from Wikipedia, where each biography is associated with a tabular infobox. The article's first sentence is treated as a reference for each biography.

In our few-shot setting, we followed one of the settings in [24] and used 100 parallel examples as the training set $\mathcal{D}_p$. Following our as-

2 Some works like Chen et al. [20] use a trained model to evaluate the human-written references' PPL. Although it provides small PPL values, the model does not directly evaluate the generated text. Therefore, we used a third-party pre-trained language model, namely, GPT-2, to evaluate the PPL of our generated text. In contrast to Li et al. [97] we did not finetune GPT-2 on our corpus. We use PPL as a measure of how fluent the generated text is as general English.

3 https://github.com/DavidGrangier/wikipedia-biography-dataset

sumption, we included another 400 examples of unlabeled input tables as $\mathcal{D}_u$. For comparison with [24], we did not use $\mathcal{D}_u$; instead, we synthesized 400 examples by recombining the table slots in $\mathcal{D}_p$. We performed this recombination for fair comparison as it did not include any new data. We validated our approach on 1000 examples and evaluated it on the standard test split.

Implementation Details To set a fair comparison with prior work for few-shot data-to-text generation [24][4], we use the T5-base model, which consists of a 12-layer Transformer encoder and decoder. We use a batch size of 20 due to GPU memory constraints. We accumulate gradients for three steps, which results in an actual batch size of 60. We adopted other details from Experiment I.

We use a different strategy to add missing slots in WikiBio due to the nature of this dataset. Unlike E2E, WikiBio contains more extended input tables. The reference is generally the first sentence of the WiKi article; therefore, not all input table slots are present in the reference sentence. Consequently, our algorithm inserts a subset of input slots. We determine this subset by co-occurrence statistics on the few-shot training dataset. As a heuristic, we select slots that occur in at least 10% tables of the dataset and are present in at least 10% output references.

selection of missing slots using co-occurrence statistics

Evaluation Metrics Similar to Chen et al. [24], we have also included BLEU scores for evaluating our generated sentences. However, from the evidence provided by [36], we still consider PARENT-W scores as the principal metric in our study. We study the hard version for semantic coverage since there is no soft version for the WikiBio dataset. Also, from our E2E experiments, we show that hard and soft coverages are generally close to each other. We also compute the coverage against the reference as WikiBio does not aim to cover every input slot.

Results We show our experimental results in Table 4.4. We replicated the model by Chen et al. [24] as they did not report the PARENT metrics for their fine-tuned GPT-2 model. Line 1–2 clearly shows that we achieved similar BLEU scores as Chen et al. [24], thus indicating the fairness of our replication. We improve the T5 model by 2–3 points in terms of PARENT Recall and F1 (Lines 3, 6–7) by using 400 unlabeled tables. The results indicate the effectiveness of our approach both in generating high-quality sentences for the data-to-text task and higher coverage of input slots (reflected by high PARENT scores). Our coverage score further confirms this claim. The lower PPL values indicate that our approach generates fluent sentences[5]. Line 9 shows another variant that determines missing slots by thresholding the cosine similarity of embeddings. The generic approach of

our approach generating fluent and high coverage sentences

4 Chen et al. [24] uses a 12-layer GPT-2 model

5 WikiBio corpus is more complex, with sentences containing quite a few proper nouns, such as the person names. Therefore, we observe higher PPL for WikiBio sentences than E2E

#	Model	#Train	BLEU	PARENT (P/R/F1)	PPL	AvgLen	Coverage (Table)	Coverage (Reference)
1	GPT2+copy [24]	p:100	29.5	–	–	–	–	–
2	GPT2+copy (our replication)	p:100	29.05	59.03 / 26.63 / 33.59	314.03	20.01	27.07%	55.27%
3	T5	p:100	35.87	**65.21** / 29.59 / 38.00	219.03	17.35	38.45%	76.61%
4	T5 self-train (Recomb)	p:100	**36.00**	64.74 / 29.58 / 37.91	219.40	17.27	38.20%	76.01%
5	T5 S&L (Recomb)	p:100	35.41	64.10 / **30.23** / **38.34**	**218.48**	18.75	**41.29%**	**76.75%**
6	T5 self-train	p:100, u:400	35.62	64.68 / 29.92 / 38.19	216.19	18.17	40.22%	76.85%
7	T5 S&L	p:100, u:400	**35.92**	64.56 / **32.28** / **40.27**	**211.35**	19.84	**42.45%**	**79.14%**
8	T5 S&L (cosine similarity)	p:100, u:400	35.44	63.63 / 31.32 / 39.36	233.18	18.92	41.28%	75.68%

Table 4.4: Test results on WikiBio (in the Humans domain). "p:" and "u:" denote the number of parallel and unlabeled training examples, respectively. "Recomb" means that we synthesize 400 examples by recombining table slots in the parallel corpus. The bold font indicates the best performance in each group that also outperforms the baselines in Lines 1–4. Coverage scores are computed against the input table and the reference text, respectively.

Model	Coverage	Fluency	Overall Quality
T5 w/ self-train	81.66%	**2.88±0.32**	2.1±0.34
T5 w/ S&L	**99.58%**	2.75±0.43	**2.81±0.39**
p-value	6.08e-24	0.00664	3.84e-22

Table 4.5: Human evaluation results on E2E. The p-values are given by two-sided Wilcoxon paired test. It only shows whether our annotated subset has collected enough evidence for drawing a conclusion or not, instead of how different two models are. We show the standard deviation, which roughly estimates if the gap is relatively large or not.

cosine similarity is different from SER (SER is specifically engineered for E2E) and does not work well compared to our verbatim matching, thus confirming the simplicity and effectiveness of our approach in alleviating the low semantic coverage problem.

our approach improves the previous sota

In comparison to the Chen et al. [24] which is the state-of-the-art few-shot learning, we use 400 extra tables. Therefore, to set a fair comparison by not using any additional data, we synthesized 400 tables by recombining the input table slots. Line 5 and Line 2 of Table 4.4 suggest that our approach improves the previous state-of-the-art model in all metrics even without an unlabeled corpus. Comparing Line 5 with Line 7, we observe that the recombination of table slots does not perform well as using additional unlabeled tables. A plausible reason is that new tables train T5 with more slot values, which is especially useful for few-shot data-to-text generation, where we only have a few hundred examples in few-shot settings. Hence, the recombination of table slots does not serve this goal. Future research can aim to address effective data augmentation for the data-to-text generation.

4.6 DETAILED ANALYSIS

We also perform a detailed analysis of the E2E dataset and its standard variant (Line 8, Table 4.3).

Human Evaluation In addition to automatic metrics, we also conducted a human evaluation to support our findings. We obtained the T5 self-train and T5 S&L outputs and selected a random subset of 50 examples. We computed the statistical significance to demonstrate the usefulness of this small subset to conclude. We employ three annotators to evaluate each table–text pair on three criteria, namely *fluency*, *coverage*, and *overall quality*. Fluency measures if the sentence is natural, clear, and grammatically correct. We ask annotators to provide each sentence a score of 1 to 3, where a score of 3 is assigned if the

Model	PARENT(P/R/F1)	InfTime	RelTime	PPL
S&L	**66.97/63.63/64.29**	**78.05**	**1x**	**160.40**
SearchInf.	65.71/60.17/61.67	113.4	1.45x	234.19

Table 4.6: Search and learning vs. search for inference. Inference time (in seconds) and Relative time were obtained by predicting the test set on a single V100 GPU.

sentence is fluent, natural, and grammatically correct. A mostly fluent sentence with minor errors gets a score of 2, and a score of 1 means that the sentence is not fluent and has multiple grammatical errors. Coverage divides the number of input table slots present in the text by the total number of input slots. Finally, we ask annotators to assign an overall quality to each sentence with a score of 3 (good quality), 2 (average quality), and 1 (poor quality). We conducted our human evaluation in a strict blind fashion, i.e., the annotators did not know the model of a generated sentence, and examples were shuffled randomly. Table 4.5 shows the results of the human evaluation.

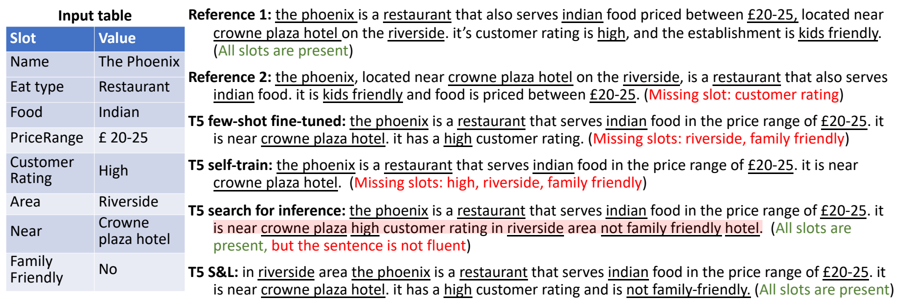
human evaluation setup

We observe that the human-annotated coverage ratio is similar to automatic counting presented in Table 4.3. Compared to a fine-tuned T5 with self-training, our S&L approach achieved near-perfect semantic coverage. Annotators did not observe any false information in both models. The results for fluency are slightly low for S&L compared to T5 self-training. However, the difference is one-third of a standard deviation, which is somewhat small compared with our improvements in other aspects. We observe that S&L has a higher overall quality than the competing method by more than two standard deviations, thus showing the effectiveness of our approach. Our human evaluation study is generally consistent with our automatic evaluation.

s&l achieving near-perfect semantic coverage

Search and learning vs. Search for inference We also perform an analysis to compare the search and learning (S&L) and search approach. In other words, we compare our approach with performing a search for inference on the test set. We present the results for this analysis in Table 4.6. We observe high performance for S&L in terms of all metrics. Specifically, the PPL values for S&L are considerably smaller than the search for inference. These small PPL values indicate the effectiveness of second-stage fine-tuning to learn from the search results for higher semantic coverage and smooth out the search noise to yield better sentences. Additionally, S&L has a better inference efficiency. Despite using the V100 GPU device and our batch implementation, the search for inference takes 45% more time than S&L in inference.

Case Study Figure 4.2 presents a case study where we have seven ground-truth references for a given data example. We have introduced two references to illustrate the existence of missing slots even



Input table

Slot	Value
Name	The Phoenix
Eat type	Restaurant
Food	Indian
PriceRange	£ 20-25
Customer Rating	High
Area	Riverside
Near	Crowne plaza hotel
Family Friendly	No

Reference 1: the phoenix is a restaurant that also serves indian food priced between £20-25, located near crowne plaza hotel on the riverside. it's customer rating is high, and the establishment is kids friendly. (All slots are present)

Reference 2: the phoenix, located near crowne plaza hotel on the riverside, is a restaurant that also serves indian food. it is kids friendly and food is priced between £20-25. (Missing slot: customer rating)

T5 few-shot fine-tuned: the phoenix is a restaurant that serves indian food in the price range of £20-25. it is near crowne plaza hotel. it has a high customer rating. (Missing slots: riverside, family friendly)

T5 self-train: the phoenix is a restaurant that serves indian food in the price range of £20-25. it is near crowne plaza hotel. (Missing slots: high, riverside, family friendly)

T5 search for inference: the phoenix is a restaurant that serves indian food in the price range of £20-25. it is near crowne plaza high customer rating in riverside area not family friendly hotel. (All slots are present, but the sentence is not fluent)

T5 S&L: in riverside area the phoenix is a restaurant that serves indian food in the price range of £20-25. it is near crowne plaza hotel. it has a high customer rating and is not family-friendly. (All slots are present)

Figure 4.2: A case study of few-shot data-to-text generation on the E2E dataset [74].

in the reference. We observe that T5 (fine-tuned with few-shot $\mathcal{D}_p$ or further self-trained with $\mathcal{D}_u$) yields fluent sentences and does not generate false information as addressed by Gong et al. [54]. However, it faces the problem of low semantic coverage, i.e., a few table slots are absent in the T5's output. We can achieve perfect slot coverage by performing a search for inference. Still, its decreases fluency like *"it is near crowne plaza high customer rating in riverside area not family friendly hotel"*. At last, our S&L approach generates a fluent sentence with high semantic coverage.

4.7 SUMMARY

We started this chapter to answer our motivating question:

Can we minimize semantic information loss when training NLG systems with minimal training data?

We introduced a scenario of few-shot learning for an NLG task called data-to-text generation. We observe that in a few-shot setting, data-to-text systems face the problem of *low semantic coverage*, i.e., important input table slots are missing the generated text. We proposed a search and learn algorithm that overcomes this problem and closes the gap between few-shot and fully supervised learning. Our approach uses the pre-trained language model to fill the missing slots and improve semantic coverage. We used the search results to fine-tune our model further, which smooths out search noise and produces fluent text. We tested our algorithm on two datasets and presented an extensive evaluation using multiple automatic metrics. We also introduced a detailed analysis of the E2E dataset to show the effectiveness of our approach, which generates fluent sentences with high semantic coverage and is efficient in terms of time efficiency. The proposed method achieves high performance on both E2E and WikiBio datasets. Specifically, we cover 98.35% of input slots on E2E, which largely alleviates the *low coverage* problem.

Chapter 3 and Chapter 4 present approaches where we use entire training datasets and, depending on the availability of unlabeled data, we either augment it using data augmentation or weak supervision. The use of entire training data is justified when the training examples are scarce, but one must pick the most informative examples during the availability of enormous amounts of training examples. It will allow us to train the models with minimum compute resources and considerably reduce the training times. In the next chapter, we will delve into this scenario, where we will design a diagnostic tool that selects the most informative training examples for training the model while ensuring minimal or no accuracy losses.

5 DATA SELECTION

In the last two chapters, we discussed the method of data generation to overcome performance drops due to the limited training examples. We used the entire training data in both scenarios because of the small data size. However, during the availability of enormous amounts of training examples, one must consider if all the training examples are helpful or not. The selection of training subset will allow us to train the models with minimum compute resources and considerably reduce the training times.

This chapter focuses on selecting the smallest subset of training data that contains the most informative training examples. We aim to use this subset for model training without degrading model performance on the downstream task. We study the problem of VQA, which is an image-conditioned NLU task. As explained in Section 2.3.1, we chose VQA due to its broad adoption in the research community and its complexity emerging from integrating two different modalities. We propose a simple diagnostic tool, called EaSe, for VQA that quantifies the difficulty of an $image, question$ example. Our diagnostic leverages the pattern of answers given by multiple human annotators for a given question. In particular, EaSe considers two aspects of the answers: their entropy and their semantic content. We prove the validity of EaSe to identify examples that are easy/hard for SOTA VQA models. We use EaSe to select increasingly difficult subsets of data, which we use to train/finetune our VQA models. We hypothesize that the difficult examples are also more informative during training. We show the effectiveness of our approach in both cases: (1) we observe that models struggle with the most difficult examples selected by EaSe; (2) training/finetuning of VQA models with only a small fraction of the most challenging examples makes these models achieve very high results, which are comparable to the results when models are trained/finetuned with the whole training data. Most importantly, we use readily available information in any VQA dataset for computing EaSe scores. We experiment on two open-source datasets and show the effectiveness of our diagnostic tool that selects the smallest training subset to recover a significant portion of model accuracy.

publication and chapter structure

The proposed method and results have been published in NAACL 2021 [73]. The rest of the chapter is organized as follows. Section 5.1 explains the problem definition. We explain our approach and the components of EaSe in Section 5.2. Section 5.3 explains the models and datasets that we used in this work. We explain complete proof-of-concept analysis in Section 5.4. We discuss experiments and results in Section 5.5 and Section 5.6. Section 5.7 discusses a complete analysis of EaSe. We provide an analysis of correlation between the EaSe scores and confidence scores provided by human annotators along with their answers. It shows that the notion of difficulty captured by EaSe is in line with that by human speakers. Finally, we conclude this chapter with a summary in Section 5.8.

5.1 PROBLEM DEFINITION

VQA [7] is a classification task that requires models to understand both visual and language modalities for the successful answer prediction. Despite massive training data and recent pre-training strategies [22, 111, 170], the challenging task of VQA still struggles to close the gap with oracle performance. There are substantial training examples in VQA datasets, like 440k training examples in one of the widely used VQA 2.0 dataset. These tremendous amounts of training examples raise the question:

Is "all you need is more training data" always true?

We aim to address this question for the task of VQA.

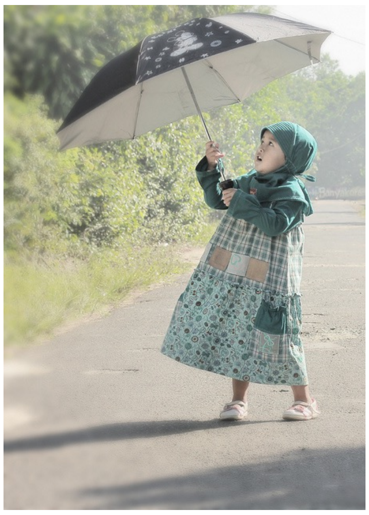
each $\langle \mathrm{image}, \mathrm{question} \rangle$ pair with ten answers

VQA datasets like VQA 2.0 [56], VizWiz [59] consists N answers by N human annotators for each $\langle \mathrm{image}, \mathrm{question} \rangle$ pair. During training a VQA model, we usually use the most frequently chosen answer as the ground-truth answer for the model. At the time of inference, the model's prediction (answer with the highest probability) is evaluated against the pattern of N ground-truth answers. The standard VQA metric [7], represented by Equation 5.1, considers the model's prediction as perfectly correct if it matches a frequent answer in the pattern and less accurate if it matches an underrepresented answer.

$$\textsc{acc} = \min\left(\frac{\mathrm{humans\ that\ said\ answer}}{3}, 1\right) \tag{5.1}$$

already available answer pattern

The metric suggests that there can be a different pattern of answers for various $\langle \mathrm{image}, \mathrm{question} \rangle$ pairs, depending on the features of the question, the image, or both. Figure 5.1 shows an example from the VQA dataset in which the annotators did not converge on one answer for either of the two questions. However, the answers to the question at the top are semantically similar (e.g., *plaid*, *plaid and floral*, etc.) while for the bottom question, all the answers are very different (e.g., *road*, *sweden*). Considering the availability of these responses, we ask ourselves if we can use this answer pattern to identify the most informative examples for training a VQA model. Therefore, our motivating question becomes:



Q: What is the pattern of the little girl's dress?
GT: plaid: 4, checks and flowers: 1, checkered with flowers: 1, polka dots, squares, plaid: 1, squares and flowers: 1, flowers: 1, plaid and floral: 1
EaSe: 1.0

Q: Where is this?
GT: road: 4, outside: 2, pakistan: 1, outdoors: 1, sidewalk: 1, sweden: 1
EaSe: 0.30

Figure 5.1: One image from VQA 2.0 with two questions and the answers by ten annotators. Frequency of each unique answer (e.g., `plaid` : 4) and EaSe values of the examples (the higher, the easier) are reported [73].

Can we use already available inter-annotator agreement to select smallest and most informative training examples?

which we aim to answer in the following sections.

We design a diagnostic tool for VQA, called EaSe, that is based on answers provided to a given question. We propose **E**ntropy **a**nd **Se**mantic content as the two main features of the answer pattern. These features are informative of the degree of difficulty of an example. Specifically, we suppose that the more scattered answer patterns are the ones with high difficulty (Figure 5.1, down). Unless some or all of those answers are semantically similar like the top $\langle question, answer \rangle$ in Figure 5.1.

5.2 APPROACH

Data selection is an important approach that aims toward efficient and cost-effective ML. Some works maximize submodular proxy functions to select diverse training examples [12, 109, 185]. In contrast, others choose corsets which is a weighted subset of training examples [79, 80, 113, 127]. Unlike the corset selection, which depends strongly on the model and training loss and does not explicitly control the validation set error, Durga et al. [43] is another work that considers validation constraints for data selection.

related work for data selection

Inspired by the usefulness of data selection approaches for designing data constrained systems, we leverage the already available inter-annotator agreements for data selection in this work. Related works have investigated these inter-annotator agreements like Gurari and Grauman [58] predicts the agreement between annotators, [191] predicts the distribution of answers for a given $\langle \mathrm{image}, \mathrm{question} \rangle$ pair. Terao et al. [172] analyze the difficulty of visual questions based on the behavior of multiple different VQA models. However, to the best of our knowledge, we are the first to leverage the inter-annotator agreement to design a diagnostic tool for data selection.

two aspects of ease: entropy and semantics

For each VQA example, an $\langle \mathrm{image}, \mathrm{question} \rangle$ pair, we aim to quantify their *difficulty*. In other words, we aim to quantify how challenging an example is for a model to provide the correct answer. We propose that we can use the (readily available) characteristics of the answer pattern provided by annotators to quantify the difficulty of the example and devise a *diagnostic* tool that builds on this assumption. We focus on two aspects of the answer pattern. First is its **Entropy** where we check how scattered an answer pattern is in terms of the number of unique answer strings. The second aspect is its **Semantics** which checks the similarity or dissimilarity of the answers in the pattern with respect to their overall semantic representation. We call our diagnostic tool **EaSe** and we explain its components in the following sections.

ENTROPY (E) We consider all the answers for a given example and use Equation 5.2, similar to Yang, Grauman, and Gurari [191], to measure the entropy of an answer pattern.

$$E(p_f) = \frac{-1}{\eta} \sum_{k=1}^{M} p_k * \log(p_k) \tag{5.2}$$

entropy captures the nature of distribution

Here, p_f represents the distribution of the M unique answers based on their frequency. η represents the highest possible Entropy value[1]. We use η to normalize E in $[0, 1]$. The highly scattered distributions get high E values (close to 1). The highly consistent distributions like the ones when all annotators agree on the same answer get low values of E (close to 0).

SEMANTICS (SE) Entropy considers the frequency of unique answer strings in a given pattern. It treats all the answer strings differently, i.e., doesn't take into account whether the answer strings are semantically similar or not. However, semantics play an important role in deciding the easiness or difficulty of an example. All annotators answering semantically different answers reveal inconsistencies among annotators. This inconsistency indicates the difficulty of an example. On the other hand, semantically similar answers are a proxy for the example's easiness, though these answers have different surface realization (e.g., *a couple* vs. *a pair*).

semantics considers semantics of answers to design new distribution

1 the maximum Entropy value is 2.302 in our data.

We use pre-trained word embeddings [124] for designing SE. We re-organize the answer pattern by aggregating semantically similar answers and their corresponding frequencies. We summarize all the steps as follows.

- For each answer in the pattern, we average its word embeddings to compute its representation, similar to Chao, Hu, and Sha [18].
- We build an average representation of all the unique answers and call it a centroid. The centroid encodes the overall semantics in the pattern.
- We compute the pairwise cosine similarity ($\cos$) between each unique answer in the pattern and the centroid. We clamp the negative values to 0 such that the similarity values are in $[0, 1]$.
- Finally, we group all the answers whose $\cos$ with the centroid embedding is greater than a certain threshold. The threshold τ is set dynamically. To adapt to the features of each data point, we compute τ at the *datum*-level. It is defined by:

$$\tau = \cos\left(\textsc{max},\ \mathrm{centroid}\right) - \varepsilon \tag{5.3}$$

Here, ε is a small positive number, close to 0. We use $\varepsilon = 0.0001$ in our experiments. The answer with the maximum frequency in the pattern is called MAX. We use the lowest τ if more than one MAX is present. At last, we obtain a new distribution such that all the answers that are semantically consistent with the centroid are combined, and we add their frequencies.

EASE DIAGNOSTIC We represent the new distribution of answers after applying SE as p_{se}. Once p_{se} is obtained, we compute EASE using Equation 5.4.

$$\mathrm{EASE}(p_{se}) = 1 - E(p_{se}) \tag{5.4}$$

ease values increasing with ease of a example

Here, the second term quantifies the Entropy of p_{se} (as in Equation 5.2), and the first term makes the value of EASE increase with the easiness of an example. We get a single value of EASE in $[0, 1]$ that quantifies the ease of a VQA example.

5.3 MODELS AND DATASET

We use two models and two datasets to experiment with our approach. The first VQA model, called BUTD [6] uses a GRU to encode input questions and attends to the image ROI features that enable region-based attention for answer generation. The second model,

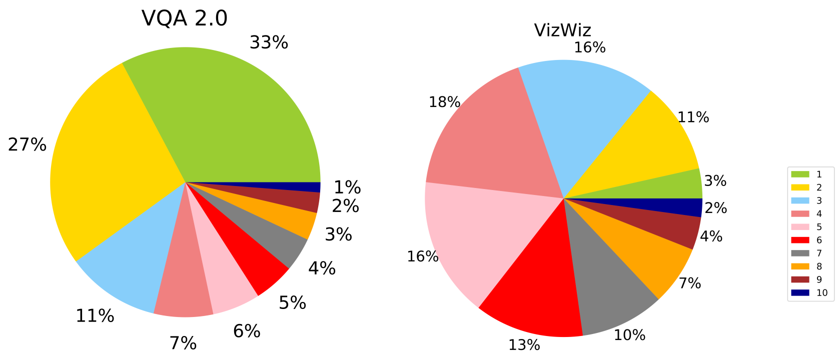


Figure 5.2: Distribution of examples in the validation splits of VQA 2.0 and VizWiz, against number of unique answers. E.g., in 33% examples in VQA 2.0, all annotators gave the same answer [73].

called LXMERT [170], is a Transformer-based architecture that is pre-trained on several vision & language tasks. We have used the default parameters as set in the original implementation. We trained BUTD and have fine-tuned LXMERT and have evaluated on the datasets described below.

experimenting with two completely different datasets

We experiment with two VQA datasets, namely VQA 2.0 [56] and VizWiz [59]. The datasets differ from each other, like object-centered images in VQA 2.0 vs. everyday-life images in VizWiz. Also, the type and purpose of their questions vary; questions in VQA 2.0 are written & crowdsourced, while VizWiz questions are spoken and goal-oriented. These differences motivate our selection for using them to see the applicability of our diagnostic tool in two completely different settings. These models and datasets are explained in detail in Section 2.3.3 and Section 2.3.2 of Chapter 2.

We perform a preliminary analysis of the answers to the questions present in the validation split. Each $\langle \mathtt{image}, \mathtt{question} \rangle$, is coupled with 10 answers provided by 10 human annotators. We use these annotations to observe the human agreement for each training example. Figure 5.2 shows the distribution of examples against the number of unique answers for both VQA 2.0 and VizWiz datasets. It clearly shows that for VQA 2.0, 33% of the questions have the same answer from all annotators. However, the percentage drops significantly for VizWiz, where only 3% of the questions have the same answer from all annotators. We use this disagreement to represent the difficulty of these datasets, where more disagreement points to more challenging examples. We refer the reader to work by Jolly et al. [72] for more details on data distribution for VQA datasets.

5.4 PROOF-OF-CONCEPT ANALYSIS

We test our hypothesis by computing the EaSe values for each example in the training and validation partitions of both datasets. Based

Method	Split	VQA 2.0		VizWiz	
		Train	Validation	Train	Validation
EaSe	TH	40522 (9%)	19805 (9%)	3201 (16%)	522 (16%)
	BH	189281 (43%)	92606 (43%)	10443 (52%)	1646 (52%)
	E	213954 (48%)	101943 (48%)	6356 (32%)	1005 (32%)
Entropy	TH	108457 (25%)	53230 (25%)	11903 (60%)	1897 (60%)
	BH	187287 (42%)	90896 (42%)	7337 (36%)	1165 (37%)
	E	148013 (33%)	70228 (33%)	760 (4%)	111 (3%)
	Total	443757 (100%)	214354 (100%)	20000 (100%)	3173 (100%)

Table 5.1: Top: Number of examples in the TH, BH, and E splits of VQA 2.0 and VizWiz based on EaSe. Bottom: number of examples based on Entropy. In brackets: percentage in the corresponding Train/-Validation partition.

on their EaSe value, we assign each example into *three* splits. The top row of Table 5.1 shows the number of examples per split. We define these splits as **easy (E)** with values for EaSe $= 1.0$, **bottom-hard (BH)** with values as $0.5 <=$ EaSe < 1.0, and **top-hard (TH)** with values of EaSe < 0.5. We test our models on each of these splits.

For the correctness of our hypothesis, we assume that models should struggle with the more challenging (harder) splits selected by EaSe. We show the performance of all models, BUTD, LXMERT, and LXMERT-S, in Table 5.2. LXMERT-S represents a version of LXMERT, trained from scratch on the task. We observe that all the models achieve low performance on the hard splits (TH), and the accuracy values are halved in comparison to the entire (*all*) data. However, high values of LXMERT show that pre-training is beneficial where the pre-trained version outperforms the non-pretrained (LXMERT-S) in both datasets and all splits, with a margin of 8 points on the complete data.

sota models struggle with the hardest split selected by ease

We also run the same analysis using Entropy (specifically, $1 - \mathtt{Entropy}$) for showing the comparison with EaSe, as shown in the bottom row of Table 5.1. We use Equation 5.2 to compute Entropy over the original answer distribution and subtract this result from 1. We use the same criteria as EaSe (Section 5.4) to divide examples into TH, BH, and E splits. We observe that the two methods give rise to very different data distributions. As we can see, Entropy assigns much more cases (25%) than EaSe (9%) to the TH split in the training partition of

completely different distributions for ease and entropy

Dataset / Split		BUTD	LXMERT	LXMERT-S
VQA 2.0	*all*	63.43	71.48	63.18
	TH	29.82	36.56	30.52
	BH	63.97	71.26	64.06
	E	69.47	78.46	68.73
VizWiz	*all*	50.35	53.75	45.79
	TH	29.48	31.84	26.61
	BH	49.08	52.82	44.38
	E	63.27	66.65	58.08

Table 5.2: Accuracy by BUTD, LXMERT, and LXMERT-S on the entire validation set (*all*) of VQA 2.0 and VizWiz and the *three* splits defined by EaSe. For all models in both datasets, accuracy consistently increases from TH to E.

Dataset/Split		BUTD	LXMERT	LXMERT-S
VQA 2.0	TH	34.73	42.2	34.89
	BH	71.31	78.66	71.22
	E	74.98	84.38	74.21
VizWiz	TH	44.40	46.79	41.48
	BH	59.25	64.02	53.31
	E	52.25	64.86	40.54

Table 5.3: Accuracy by BUTD, LXMERT, and LXMERT-S on three validation splits of VQA 2.0 and VizWiz. The splits are obtained via Entropy.

VQA 2.0. Further, Entropy assigns a few examples to the Easy split (33% Entropy vs. 48% EaSe) for training partition of VQA 2.0.

crucial role of semantics in determining ease scores

The comparison provides two important insights. Firstly, it confirms the crucial role of our semantic component in determining the EaSe scores. Secondly, we observe that for the three splits defined by Entropy, the results obtained by the three models follow a less clear pattern compared to the ones obtained by EaSe. We present these results in Table 5.3. Here, both BUTD and LXMERT-S achieve higher results in BH in comparison to E for the VizWiz dataset. It indicates that Entropy is not as effective as our tool to measure the difficulty of an example.

Finally, we also performed an experiment that tests model performances on splits of the same size as that of EaSe's TH, BH, and E but

Dataset/Split		BUTD	LXMERT	LXMERT-S
VQA 2.0	TH	63.42	71.56	63.10
	BH	63.45	71.43	63.16
	E	63.35	71.46	63.17
VizWiz	TH	50.31	53.59	45.79
	BH	49.88	53.31	45.97
	E	50.26	53.77	46.18

Table 5.4: Accuracy by BUTD, LXMERT, and LXMERT-S on three random splits of validation data of VQA 2.0 and VizWiz. The random splits are of same size as that of TH, BH, and E as mentioned in Section 5.4.

including random examples. Table 5.4 shows the results where all the splits have almost the same accuracy. Unlike our proposed diagnostic EaSe, which selects the most difficult subset (TH), which is hard for all models, we don't observe any such pattern in the random split selection. In other words, we regard no difference in performance between the three splits. We performed the sampling 10 times and reported the averaged results.

other methods do not show any pattern

The proof-of-concept analysis provides insights into the limitations of current SOTA models, including extensively pre-trained LXMERT, to perform well on the hard splits deemed by EaSe. The analysis suggests that our diagnostic tool genuinely selects the most challenging examples of a dataset.

ease determines the most challenging training examples

Motivated by the promising results by EaSe to select the smallest & most challenging subset, we then ask ourselves, *can we make our models robust by training with these hard examples?* Our intuition is that challenging examples could be more informative during training compared to the easy examples. We test our hypothesis in the following sections, where we use the splits defined by EaSe to train models in a HardFirst (HF) approach.

5.5 EXPERIMENTS

As the name suggests, we train models incrementally in a hard first approach. We start by only using TH examples, then adding BH examples, and finally using all training examples (TH+BH+E). We initialize the weights for the first stage randomly, and for each incremental stage, we load the model weights from previous stages. For VQA 2.0, we have 9.13% (TH) examples for first stage, 51.79% for second stage (TH+BH), and 100% (ALL) for third stage. The percentages for VizWiz are 16%, 68.22%, and 100% for the respective stages. We hypothesize that harder splits, with low EaSe scores, possess rich multimodal

Model	TD	VQA 2.0				VizWiz			
		all	TH	BH	E	*all*	TH	BH	E
BUTD	TH(R)*	50.14	20.46	53.34	53.0	42.75	24.91	40.57	55.58
BUTD	TH	44.13	26.1	51.3	41.13	42.46	25.1	39.69	56.02
BUTD	TH+BH	56.6	29.73	61.2	57.64	48.58	29.58	47.57	60.1
BUTD	TH+BH+E	61.43	29.61	62.81	66.36	50.12	29.56	48.95	62.73
LXMERT	TH(R)*	69.61	34.76	69.44	76.55	46.42	26.03	45.78	58.06
LXMERT	TH	67.24	35.64	67.58	73.02	46.65	26.13	45.79	58.73
LXMERT	TH+BH	69.85	37.05	70.63	75.52	51.65	30.29	50.07	65.36
LXMERT	TH+BH+E	70.57	35.51	70.26	77.65	53.40	32.82	52.26	65.97

Table 5.5: Accuracy on each split of VQA 2.0 and VizWiz obtained by gradually training models first on TH, then adding BH and finally adding E examples. TD refers to type of training data used for training. TH(R) refers to the setting in which we use a split randomly sampled from the training data with the same size of TH. *The random sampling was performed 10 times; as such, the reported accuracy is the average over 10 accuracy values.

information, which could be informative during a model's learning. We also evaluate models in the TH(R) condition for fair comparisons. Here, we train/finetune models with a random subset of training data that has the same size as that of TH. We repeat the sampling 10 times and report the average accuracy results.

5.6 RESULT DISCUSSION

Table 5.5 shows our experimental results. We observe that the experiments support our hypothesis. Here, in comparison to the model trained on whole data Table 5.2, the BUTD model obtains 90% of all validation accuracy using only 52% of training data (TH+BH) for the VQA 2.0 dataset. The results are more pronounced in VizWiz, where 68% of the total data (TH+BH) leads to comparable performance as the one obtained with the whole training data. We observe similar results for LXMERT as LXMERT achieved 97% of validation accuracy with 68% of training data in VizWiz. For VQA 2.0, LXMERT achieves 98% of validation accuracy using only 52% of training data.

models recover significant accuracy gains using half the size of original training data

In comparison to the TH(R) condition, we observe that models trained/finetuned with the TH split achieve higher results in the TH split of validation sets for both datasets. It confirms that TH examples are valuable in dealing with challenging cases. Furthermore, we observe that during the evaluation of the entire data (*all*), the models perform similar to TH(R) for VizWiz and slightly worse than TH(R) for VQA 2.0. However, the results are not surprising as during sam-

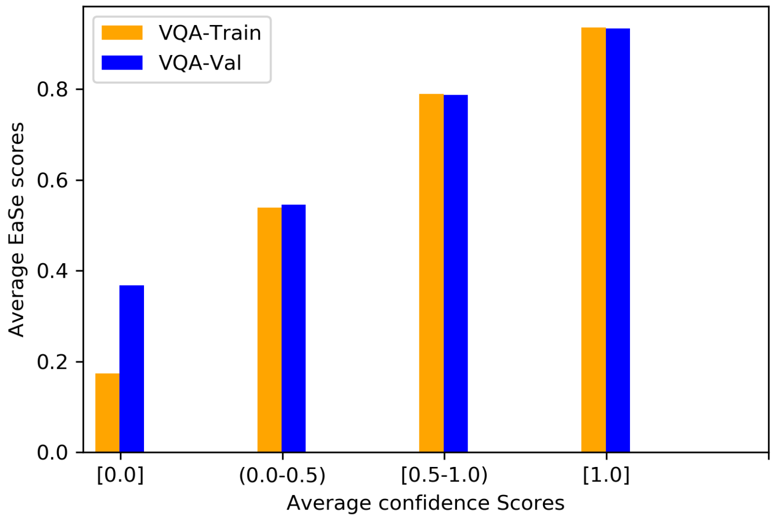


Figure 5.3: Average EaSe scores against binned confidence scores in VQA 2.0. Closed/open brackets indicate that values are included/not-included in the bin [73].

pling form VizWiz 68% cases are either BH or TH, and therefore it will likely produce a more similar distribution to that of TH. Nevertheless, there are 48% of E cases in VQA 2.0, thus leading to a sampling of more similar distribution to that of E. Due to the same proportions in the validation set of VQA 2.0, training/finetuning with the easier cases will have a positive impact on E, thus driving the performance on *all*. These results indicate that the hard examples by EaSe are more informative than the easier ones and help models to obtain comparable performance with significantly less training data.

5.7 ANALYSIS OF EASE

5.7.1 EaSe *vs. Confidence Scores*

We use the confidence scores provided by the annotators along with their answers to test whether EaSe correlates with human intuition of *difficulty* while answering a question. The confidence scores evaluate whether the annotators are confident while providing an answer, and the values are *yes*, *maybe*, *no*. We map these values as *yes*, *maybe*, *no* to 1, 0.5, and 0, respectively, and compute the average confidence score for each example. We compute *Spearman's* correlation between confidence scores and EaSe scores, and find substantial positive correlation in both training ($\rho = 0.49$) and validation ($\rho = 0.48$) sets. Figure 5.3 shows the trend in VQA 2.0 dataset, where we have higher confidence scores corresponding to increasingly high EaSe values. We observe a similar trend for VizWiz, Figure 5.4, which shows that EaSe

high correlation between ease and human intuition of difficulty

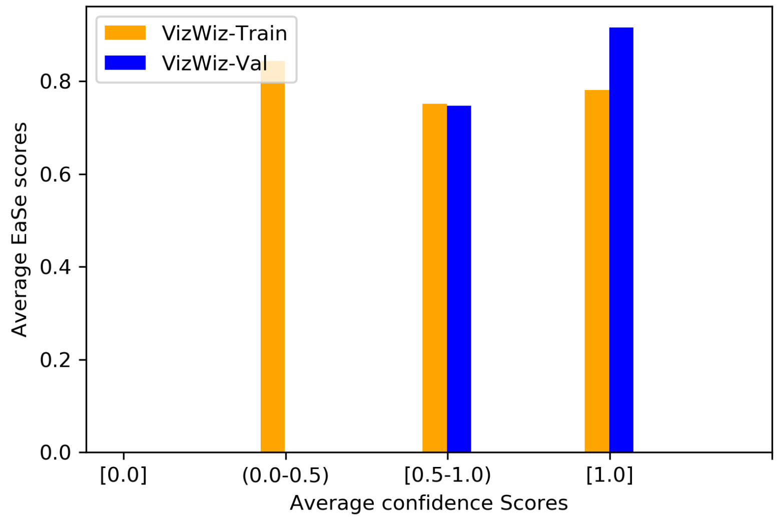


Figure 5.4: Average EaSe scores per confidence scores provided by annotators for both splits of VizWiz. Open/close brackets indicate that values are not/ included [73].

correlates with human intuition of having difficultly which answering a question.

5.7.2 EaSE *vs. Question Types*

ease selects difficult question type for its hard split

In this analysis, we study the question types present in the hard splits selected by EaSe. Our intuition is that EaSe must contain question types that are challenging for VQA models like *wh-* and count questions. Figure 5.5 and Figure 5.6 shows the results of this analysis. We observe a high proportion of count (*Number*) and *wh-* (*Other*) questions in the hardest split as compared to the other splits of VQA 2.0. We observe a similar pattern for VizWiz in Figure 5.6, where there are a higher number of Other question types in the hardest split selected by EaSe. Furthermore, there are very few Unanswerable questions in TOP-HARD split which show an interesting property of EaSe that doesn't consider the usual notion of associating Unanswerable questions with hard rather; it looks at human agreement/disagreement to determine the difficulty.

5.8 SUMMARY

We started this chapter to answer our motivating questions:

Is "all you need is more training data" always true?

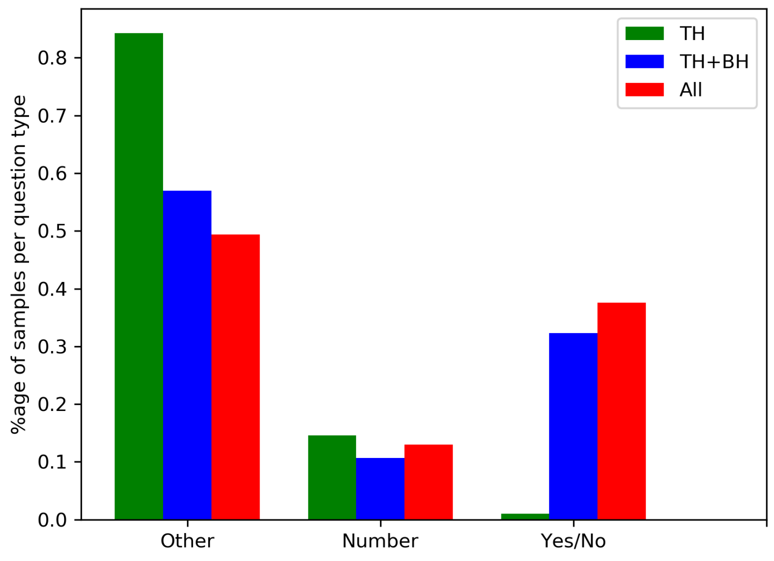


Figure 5.5: Percentage of examples per question type in VQA 2.0-train for each of the three splits used in the HF training regime. *Other* contains all *wh-* questions, *Number* count questions, *Yes/No* polar questions.

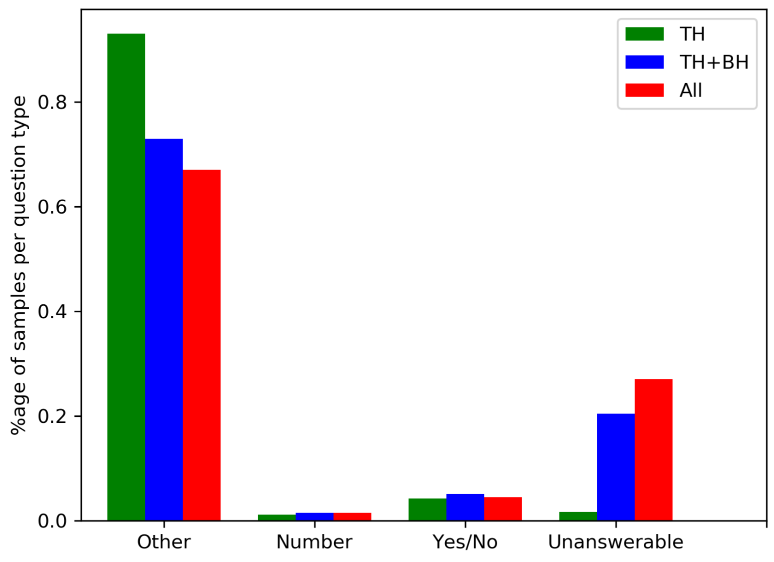


Figure 5.6: Number of examples per question-type in VizWiz-train for each of the three splits used in HF training regime. Here, Other belongs to reasoning questions (why, which, where), Number to counting questions, and Yes/No to polar questions.

Can we use already available inter-annotator agreement to select smallest and most informative training examples?

We presented a diagnostic tool for data selection that allows the training of VQA models with limited training examples without facing any accuracy degradations. This way, selecting the most informative subset of training data will allow us to train the models with minimum compute resources and will considerably reduce the training times. We studied the problem of VQA and leveraged the already available inter-annotator agreements to select the most informative training examples. Our tool, called EaSe, is based on the entropy and semantic similarity of answers provided by human annotators for each training example. We presented a proof-of-concept analysis that shows the effectiveness of EaSe in selecting the most challenging split of data which is difficult for the SOTA models. We experimented with two datasets and used two VQA models. We showed that for both datasets, both VQA models recover a significant portion of validation accuracy by using the hard splits (selected by EaSe) during training.

This completes the third part of our thesis, where we presented the role of data in handling data constraints. We discussed approaches to generate synthetic data and select a subset of training data for efficient model training. It's worth noting that these approaches can extend and generate only task-specific training data. However, as mentioned in Section 2.4, one may face problems like different distribution for training and test splits, where the above approaches are not applicable. In the next part, we will investigate the method of exploiting latent representations and show how we can leverage them for the scenarios mentioned above. We will continue our study on the same VQA 2.0 dataset. We will understand the relationship between two question types and highlight the possibility of designing future datasets with limited costs, i.e., enabling a data-constrained VQA.

Part IV

EXPLOITING LATENT EMBEDDING SPACE TO HANDLE DATA CONSTRAINTS

6 FEATURE SPACE LEARNING

This chapter presents the proposed methods to investigate the feature embedding space to handle data constraints. We study the widely used VQA 2.0 [56] dataset for our analysis. VQA 2.0 contains two question types, namely, yes/no (polar) and non yes/no (non-polar). We conduct an empirical study to understand the feature space of these two question types for VQA models. We observe that 38% of the questions have two answers ("yes" and "no") while the remaining 62% questions belong to 3127 answers. Such an over-representation of polar questions raises an important question *Are there any sources of bias emerging from the over-representation of polar questions?* In our experiments, we addressed this question and measured the confounding factors when VQA models are trained using both question types. We also performed an experiment called cross-polarity evaluation to analyze the alignment of polar and non-polar feature spaces. Our results of exploring the intermediate feature space of visual-text embeddings show that the feature space of polar questions encodes sufficient structure to answer many non-polar questions. The results indicate that the two feature spaces are strongly aligned, hence the expression $\mathbf{P} \approx \mathbf{NP}$. Therefore, in a data-constraint setting, one can use polar questions, which are easier to annotate via crowd-sourcing, for building future VQA datasets.

publication and chapter structure

The methods and models presented in this chapter have been published in ICPR 2020 [71]. We discuss problem definition in Section 6.1. Section 6.2 explains our method. Section 6.3 explains the competing methods. We summarize results and a detailed discussion in Section 6.4 and Section 6.5. Finally we conclude this chapter with a summary in Section 6.6.

6.1 PROBLEM DEFINITION

As discussed in Chapter 5, Visual Question Answering [7] is a classification task that requires machine learning models to jointly learn from image and textual modalities for a successful answer prediction. Since the release of VQA by Antol et al. [7], VQA models were seen to employ dataset biases instead of understanding images or questions. For, e.g., a VQA model for questions starting with "*what sport...*" usually gave "*tennis*" as an answer since the answer was seen more often in training data for VQA 1.0 [7]. These uneven data distributions lead to the introduction of large VQA datasets [56], new metrics [77], regularizers [3] or re-balanced partitions for existing datasets [3, 4].

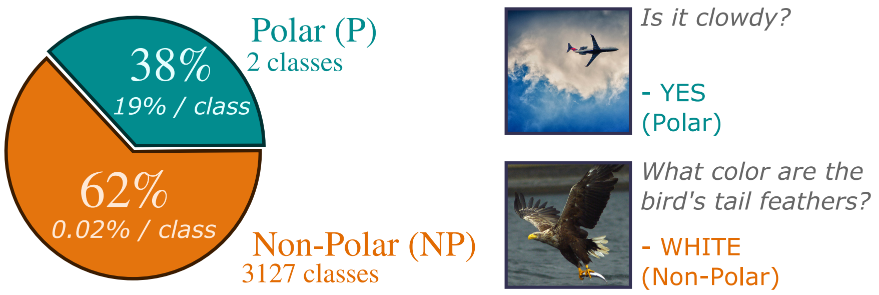


Figure 6.1: Distribution of polar (P) and non-polar (NP) examples in VQA 2.0 [71].

We analyze a different source of bias that emerges from joint training of two question types where one type is significantly dominant in the training data. VQA datasets like COCO-QA, VQA 1.0, or VQA 2.0 contain two question types, which we call polar and non-polar. Here, polar questions have an answer, "yes or no", and non-polar questions are those whose answers are other than "yes or no" (counting, wh-questions, etc.). Figure 6.1 shows the distribution of these questions in one of the widely used VQA datasets. We observe that polar questions comprise 38% of the total questions, thus leading "yes or no" to appear for 19% of the time. On the other hand, each remaining 3127 answers appears 0.02% time.

VQA models are trained jointly, treating each unique answer independently, despite this class imbalance. The SOTA models don't mention any balancing techniques like class regularization or mini-batch resampling [103]. The imbalance can lead to significant performance issues, like models allocating more capacity to answer polar questions due to their over appearance during training. Therefore, we focus on measuring the impact (positive or negative) stemming from the over-representation of polar questions in VQA datasets. We study if there are any confounding factors between polar and non-polar questions when projected into a common feature space? Our motivating question then becomes:

understand over-representation of polar questions

Can we exploit feature embedding space to overcome dataset constraints?

which we aim to answer in following sections. We perform a series of experiments on a high-performance VQA classifier. In our analysis of data distribution changes, we observe considerable overlap between features from polar and non-polar questions. The overlap favors the overall optimization objective such that we can use polar features to answer non-polar questions (and vice-versa).

6.2 METHODS

Researchers have investigated polar/non-polar questions in VQA, like Zhang et al. [197] highlighted the importance of balanced polar questions i.e., same number of questions with"*yes*" and "*no*" as an answer. On the other hand, Suhr et al. [167] released a polar-only dataset to estimate the understanding of logic reasoning by the VQA system. In contrast to our study, all these works have studied the effects of imbalance within the polar space in isolation, thus disregarding the interactions between polar and non-polar questions in a joint feature space. Teney et al. [171] presented good practices for training models using VQA 2.0 in which they ensured a balance between examples with "*yes*" and "*no*" answers. Still, there is no consideration of the balance between polar and non-polar questions. Agrawal et al. [3] showed an explicit separation of polar and non-polar questions for the GVQA model. They processed these two question types separately and used the polar questions to verify for the contained non-polar concepts.

previous work regarding polar/non-polar questions

As discussed in Chapter 1, human annotators are employed for data collection, and since polar questions are more accessible to obtain than the non-polar ones, datasets end up with a distribution of question types heavily skewed towards polar questions. Synthetically generated datasets like CLEVR [68] balance such over-representation with a more uniform distribution along with different answer types. However, both natural and synthetic datasets have entirely dismissed the use of polar questions to alleviate such over-representation issues [201] and have attained more complex questions [86]. In this work, we study the influence of polar questions on the non-polar counterparts during their joint training, a conventional regime for modern VQA models.

We used VQA 2.0 **dataset** for all experiments. As mentioned in Chapter 2, VQA 2.0 contains 443757 training examples and 214354 for validation. Like related work, we use the training data for training and report our accuracy on the validation data. VQA 2.0 is one of the largest non-synthetic corpora for VQA, containing polar and non-polar questions. We use this dataset because of its property of having a uniform distribution between questions with "*yes*" and "*no*" as ground-truth. Also, it has an adjusted distribution of non-polar questions w.r.t./ VQA 1.0. We split VQA 2.0 into two disjoint sets that contain polar and non-polar questions. It enables us to assess the upper bound of a VQA model when the model uses only one or the other question type.

experimental setup

We use the well-known BUTD model [6] for our experiments. Figure 6.2 represents the architecture of our model. There are three modules; an image embedding, a text embedding, and a joint classification module. Both image and text embeddings are projected to a 512-dimensional space to model joint visual-text space and are

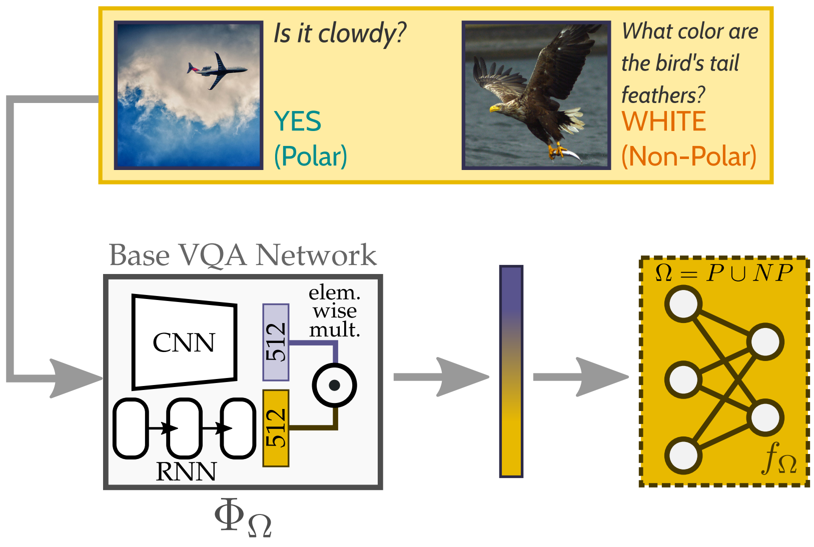


Figure 6.2: Overview of the VQA model used throughout this paper. This is an re-implementation from the winning entry of the 2017 VQA challenge. It is composed by two main modules: a textual-visual joint embedding (base VQA model denoted as Φ_Ω) and a shallow 2-layer classifier (denoted as f_Ω) Jolly et al. [71].

fused through an element-wise product. The fused or joint embedding passes through a sequence of fully connected layers such that we normalize the output by a softmax operation.

We use separate names for different parts of this VQA model in this work. We call the first part of the network, up until the point-wise multiplication of the 512-dimensional projection of multimodal space, as the **base VQA network** and denote it as Φ. The second part, comprising two fully connected layers and the output layer, is referred to as **the classifier** and is denoted as f. We refer the prediction by the network as $\hat{y} = f(\Phi(x))$. We use Φ_P and f_P to refer to corresponding modules when only polar questions are used for training the model. Similarly, Φ_{NP} and f_{NP} refer to modules that use only non-polar examples for training. We also trained our model using both polar and non-polar examples for completeness and refer to the corresponding modules as Φ_Ω and f_Ω.

6.3 COMPETING METHODS

We performed three main experiments summarized in the following sections.

6.3.1 *Baseline*

We train the entire VQA model on the complete dataset for our baseline. We did not make any considerations regarding polar and non-

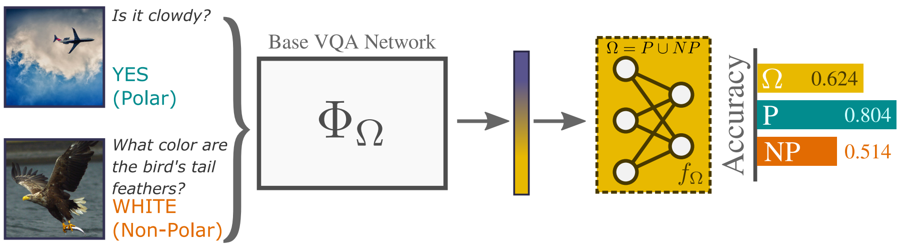


Figure 6.3: Outline of the baseline experiment: The full VQA model is trained on the full VQA 2.0 training set. Accuracy is reported on the full validation set Ω, on the polar questions in the validation set P and on the non-polar counterparts NP [71].

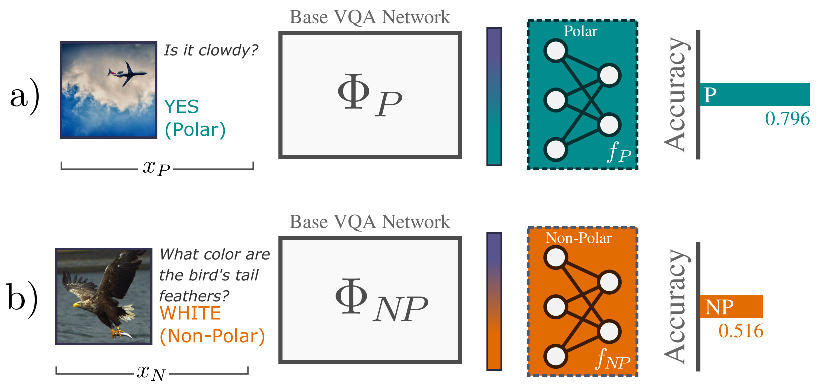


Figure 6.4: Unbiased Upper Bound experiment: two copies of the same architecture for the baseline is used. One copy is trained only on polar questions and the second copy is trained only on non-polar questions. Accuracy for both is reported independently [71].

polar questions. Similar to the original work [6], we use training examples whose answers appear at least eight times in the entire dataset. This way, there is an answer space of 3129 answer classes, two of which are "*yes*" and "*no*" (polar answer space). We did not perform any pre-training using the Visual Genome dataset and used 36 regions of interest for image embedding. We use RELUs as the activation function and train the model using Adamax [82] optimizer. We use a learning rate of 2×10^{-3} on the complete training set and report the standard VQA accuracy for the entire validation split of VQA 2.0. We also report accuracy for the polar and non-polar subsets of the validation data, where the corresponding subsets contain only polar or non-polar questions. Figure 6.3 represents our baseline. Our baseline serves as a reference to quantify the impact of polar and non-polar questions when both are used to train the VQA model. In the following experiments, we will study the impact of each of these types separately.

training model using entire training data

6.3.2 *Unbiased Upper Bound*

The experiment aims to obtain an empirical upper bound of the model such that the issue of over-representation does not play any role. We train two separate versions of the same model from scratch; the first is trained using only non-polar questions, and the second uses only polar questions for training. We use $f_{NP} \circ \Phi_{NP}$ and $f_P \circ \Phi_P$ to refer to these two models. We report the corresponding VQA accuracy for both of these models in Figure 6.4. Here, $f_{NP} \circ \Phi_{NP}$ is evaluated on the non-polar ones and $f_P \circ \Phi_P$ on the polar questions of the validation set.

training model separately using polar and non-polar examples

The experiment allows us to compare the capacity of the same VQA model with the one used as the baseline when it deals with only one kind of questions. In the current training scenario using either polar or non-polar questions, the network uses 100 % of its capacity to extract the necessary semantics for the corresponding question type. There is no burden of modeling features to distinguish polar and non-polar questions. Therefore, the current setting avoids any bias arising from the imbalance of polar and non-polar questions. Hence, we expect both variants accuracy to be higher than their corresponding values for the baseline experiment. However, if small or no deviations arise w.r.t. the baseline, we can conclude that there is no adverse effect on the baseline VQA due to confounding factors between polar and non-polar questions.

6.3.3 *Cross-Polarity Evaluation*

The above experiment studies the potential confounding factors between polar and non-polar examples. In this experiment, we study the overlap of the distribution of polar questions with non-polar ones. We perform an experiment using **transfer learning** as the feature projections from Φ_P and Φ_{NP} share the same dimensional space. Figure 6.5 shows the experiment outline. We use Φ_P and Φ_{NP}, from the previous experiment, as fixed pre-trained feature extractors. We train two new classifiers that use features from the fixed feature extractor module. Firstly, we train a new polar classifier f_P that uses features from the fixed pre-trained non-polar module Φ_{NP}. Similarly, we train a new non-polar classifier f_{NP} that uses features from Φ_P. For fairness, we use the same architecture for f_P and f_{NP} as that of the previous unbiased upper bound experiments.

training polar/non-polar classifier using features from pre-trained non-polar/polar feature extractors

Cross-polarity evaluation allows us to understand the descriptiveness of the feature space of polar questions to answer the non-polar ones and vice-versa. Theoretically, it is possible at a semantic level since we can use polar structure to ask non-polar questions. E.g., a polar question *"Are the bird's tail feathers white?"* represents a non-polar question *"What color are the bird's tail feathers?"* with answer *"white"*.

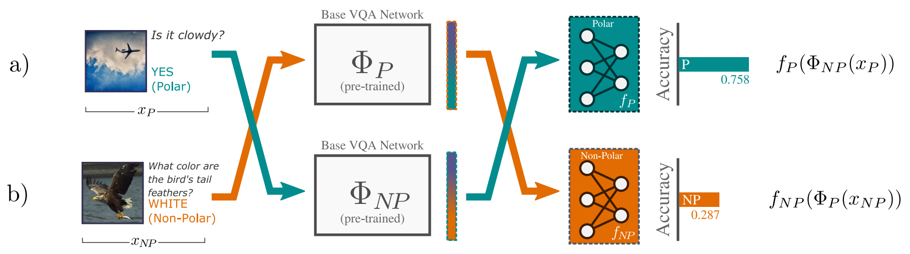


Figure 6.5: Cross-Polarity evaluation: a) We project polar inputs x_P using a pre-trained base network Φ_{NP}, which has only seen non-polar samples during its training, into non-polar feature space $\Phi_{NP}(x_P)$ and train a shallow polar classifier $f_P(\Phi_{NP}(x_P))$ on this representation. b) Vice-versa. Intuitively, the experiment measures the extent by which non-polar questions can be answered based on features extracted from a polar space and vice versa [71].

Considering the large space of non-polar concepts that covers more than 3000 classes, it is intuitively expected that Φ_{NP} represents a rich enough structure that can be condensed and reused to answer polar questions, which are only two classes. However, the usage of polar questions to yield a rich embedding space to answer a wide spectrum of non-polar concepts is unexpected.

6.4 RESULTS

Table 6.1 presents the results of our experiments, explained in Section 6.3. In the columns from left to right, we show the experiment name, the subset (P: Polar; NP: Non-polar; Ω: All) of the set of training examples used to train the corresponding module of the whole VQA ensemble (Φ or f). We show validation set accuracy and individual accuracy for polar and non-polar validation subsets. We assume Φ to be pre-trained and fixed for the cross-polarity experiments and train from scratch the corresponding f. We observe that the accuracy for the upper bound experiment is almost within the range of the baseline. Here, for a system trained on non-polar questions, the upper bound results increase 0.2 pp. On the other hand, a drop of 0.8 pp is observed for training with polar questions. When we use them together for training, these fluctuations between the upper bound and the baseline are negligible, indicating no confounding factors between polar and non-polar examples. We see that polar questions are rarely confused with any non-polar choice (Table 6.2). Further, non-polar questions are also not mistaken for any of the two polar answers. We present an in-depth analysis in Section 6.5.

no confounding factors between polar and non-polar examples

We observe a slightly different behavior in the cross-polarity experiments. The results for using a polar classifier f_P based on non-polar features from Φ_{NP} are almost as high as using Φ_P. There is an accuracy difference of 3.8 pp between cross-polar model $f_P \circ \Phi_{NP}$ and the

Task	Model		Input	Accuracy
	Φ	f	x	
Random Choice	–	–	P	0.5
	–	–	NP	0.0003
Baseline	Ω	Ω	Ω	0.624
			P	0.804
			NP	0.514
Upper bound	P	P	P	0.796
	NP	NP	NP	0.516
Cross-Polarity	NP	P	P	0.758
	P	NP	NP	0.287

Table 6.1: Summary of experimental results. The second column indicates the data used to train each of the VQA modules Φ and f. The column "*Input*" indicates the data used during evaluation of the ensemble $f(\Phi(x))$, and the column "*Accuracy*" reports the corresponding single-model VQA accuracy [7] from the validation set.

polar upper bound $f_P \circ \Phi_P$. Considering both results to be higher than 75% accuracy, we find this difference low, especially when compared to the random selection probability, which lies at 50%. On the other hand, the use of polar-only feature space to classify non-polar questions ($f_{NP} \circ \Phi_P$) shows an accuracy of 28.7%. The result is lower than the upper bound $f_{NP} \circ \Phi_{NP}$ by -22.9 pp. However, it is still notable above the random chance of 0.03 %. We discuss them further in the following section.

6.5 DISCUSSION

We analyze the results from our experiments and their implications, in this section, in the context of joint space shared by polar and non-polar features.

Firstly, we analyze the results of baseline experiments against their corresponding empirical upper bound for both polar and non-polar features. The results are indistinguishable, indicating that the model can simultaneously cope with both question types (trained with non-polar and overrepresented polar examples together) without compromising its performance. Therefore, there are no measurable confounding factors (i.e., source of bias) when we simultaneously use polar and non-polar questions. There are two possible scenarios to understand better the distribution of polar and non-polar questions in the joint feature space Φ_Ω. First, each distribution occupies a different

		Predicted Answer	
		P	NP
True Answer	P	37.59	0.11
	NP	0.77	61.53

Table 6.2: Confusion matrix of predictions for the baseline model, grouped by polarity (all numbers in percent). Polar predictions are rarely confused by any of the non-polar alternatives and vice versa.

(disjoint) sub-region of the feature space. Or there is an overlap between two feature spaces, i.e., they model (at least partially) the same semantic concepts. We identify the verification of which of the two conjectures is valid in our following analysis of the remaining experiments.

Our second observation comes from the first cross-polarity experiment. We train a polar classifier, a classifier on polar questions, by first projecting them to a feature manifold of non-polar concepts. We then measure the possibility of answering polar questions using the feature space of non-polar concepts. Since non-polar questions cover the vast complexity of topics[1], the scenario becomes intuitively simple. Also, we increase the chance level of the classification problem from $\frac{1}{3127}$ to $\frac{1}{2}$ by reducing the number of classes from 3127 to 2, which makes this problem relatively easy. Hence, there is no surprise in the observed result. The minor performance drops show that we can use non-polar feature space representation to answer most polar questions. The occurrence of the non-polar concept (e.g., "*green*" or "*bicycle*") in the polar question embedding makes answering polar questions relatively straightforward, which is achievable even by a simple classifier.

non-polar feature space answers most polar questions

Our third and the most interesting observation comes from the other direction of cross-polarity experiments. We trained a non-polar classifier, a classifier on non-polar questions that uses polar feature space. Since we are now going up from 2 to 3127 classes, our problem has become quite challenging. Also, it is unclear how much we can use the polar feature space to express the intricacies of non-polar questions. Therefore, the non-polar classifier based on features from a polar embedding space is expected to perform poorly. As mentioned in Table 6.1, the reported accuracy for this setup is 28.7%, which is lower than the upper bound of 51.6 % but is significantly higher than the random chance of $\frac{1}{3127}$. The behavior suggests that we can answer a notable subset of non-polar questions with high accuracy based on the feature space of polar questions. Does it then raise a question: *How can we find the subset of non-polar questions that can be answered*

polar feature space answers non-polar questions

1 We have 3127 answer classes in our experiments that represents non-polar concepts.

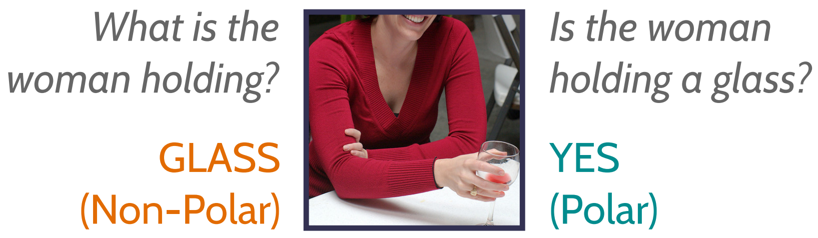


Figure 6.6: Polar questions can be used to answer non-polar questions with high accuracy as long as the polar questions relate to an existing non-polar concept. Given the image in the center, a polar question (right) and a non-polar question (left) can be asked about a common non-polar concept, namely “*glass*” [71].

using the polar feature space? which we will answer in the following sections.

Furthermore, the cross-polarity experiments show that polar feature space covers non-polar questions with numeric answers (we observe high accuracy of $f_{NP} \circ \Phi_P$ for numeric labels such as “0”, “1” and “2”). Therefore, we theorize that a general alignment exists between polar questions *about* non-polar concepts and the corresponding non-polar questions.

6.5.1 *Polar Questions About Non-Polar Concepts*

Figure 6.6 shows an example where both the non-polar question “*What is the woman holding?*” and the polar question “*Is the woman holding a glass?*” refer to the same semantic concept, namely “*glass*”. For detailed analysis, we count polar questions that talk about each of the non-polar concepts, and we focus on the non-polar concepts that frequently appear in polar questions. We call these non-polar concepts/answers “well-covered” answers. We refer to non-polar examples as $\mathcal{X}_{NP,t}$ and check the occurrence of their answers in polar questions between t_0 and t_1 times. Here, $t = [t_0, t_1]$ is a discrete interval $0 \leqslant t_0 \leqslant t_1 \leqslant T$; T is the maximum amount of occurrences that a non-polar answer is found among polar questions.

We select the polar questions in which any of the 3127 non-polar answers occur textually using the regular expression to populate the subset $\mathcal{X}_{NP,t}$. The regular expression looks for exact matches of the NP expression as long as it is surrounded by non-alphabetic characters, including BOL and EOL. Following, we count the number of polar questions that match each non-polar answer with replacement and sort them in descending order w.r.t the number of occurrences). Figure 6.7 shows the resulting histogram. The x-axis represents the 3127 non-polar answers, and the y-axis represents the number of polar questions that match the non-polar answer. We observe that 73.9% of non-polar concepts are matched by at least one polar question, 26.1% are matched in at least 30 polar questions (per non-polar con-

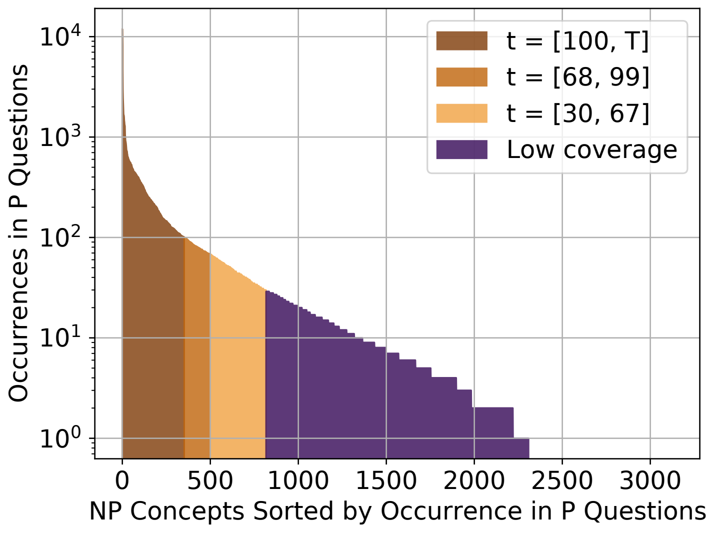


Figure 6.7: Histogram of non-polar concepts (labels) that occur between t_0 and t_1 times in polar questions [71].

Task	Model		Input	Accuracy
	Φ	f	t	
Cross-Polarity	P	NP	$[30, 67]$	0.214
	P	NP	$[68, 99]$	0.317
	P	NP	$[100, T]$	0.408

Table 6.3: Accuracy of the cross-polarity model $f_{NP} \circ \Phi_P$ only over non-polar classes which occur within polar questions between $t = [t_0, t_1]$ times i.e., $\mathcal{X}_{NP,t}$.

cept), and only 11.3% non-polar concepts are matched in at least 100 polar questions.

We test the model $f_{NP} \circ \Phi_P$, used in the last experiment (explained in Section 6.3.3), w.r.t. sets of non-polar examples that are increasingly well-covered by polar examples. Specifically, we test on $\mathcal{X}_{NP,t}$ for $t = [30, 67], [68, 99]$ and $[100, T]$. Table 6.3 shows the results of this experiment where results for well-covered non-polar concepts (40.8% for $t = [100, T]$) are considerably higher than those from the cross-polarity experiment defined in Section 6.4, where we achieve 28.7% accuracy for evaluating all non-polar classes. It clearly shows that the VQA model exhibits a positive correlation between the accuracy of non-polar classes and the number of polar questions that mention those classes. Furthermore, the results are substantially closer to the upper bound accuracy (Section 6.3.2) for non-polar questions. Instead of the original difference of 22.9p.p., it is only 9.8p.p. below. On the other hand, non-polar questions with a lower number of matching po-

accuracy increases from 28.7% to 40.8%

lar examples show lower accuracy than the cross-polarity experiment. The results indicate that we can answer non-polar questions using a feature space based on polar examples. The caveat is that the set of polar questions used to train the VQA model should convey enough semantics about the corresponding non-polar questions.

In the light of the results, a logical argument is that modeling non-polar questions with polar feature space depends on polar questions that deal with the corresponding non-polar concepts. Therefore, we conclude that polar feature space *can* carry an equivalent semantic value as the non-polar feature space, hence P $\approx$ NP.

6.6 SUMMARY

We started this chapter with our motivating question:

Can we exploit feature embedding space to overcome dataset constraints?

We studied the over-representation of polar questions in VQA datasets and its relationship with non-polar questions for the widely used VQA 2.0 dataset. Our experiments observe that the over-representation of polar questions does not introduce any biases during joint training of these two question types. We conducted a cross-polarity evaluation to study the relationship between polar and non-polar feature space. We observed a clear correlation between the distribution of polar and non-polar feature embeddings. We showed that we could use polar questions to answer non-polar questions; if polar questions used for training refer to the semantic concepts being considered in the non-polar questions. Our findings conclude that polar features (P) and non-polar features (NP) provide a rich semantic structure; thus, the expression P $\approx$ NP refers to this alignment. The usefulness of a feature space based on polar samples for answering non-polar questions is surprising and potentially ground-breaking because it can change the way of compiling future datasets. Considering the reduced costs of collecting polar questions (in comparison to non-polar ones), one can use polar questions to augment future datasets, thus enabling the designing of VQA systems in a data-constrained setting.

This chapter presents insights from the feature space of two question types within the same dataset. Following our study on exploiting latent embedding space, we will explore another scenario where VQA models are seen to struggle on question rephrasings at test time, i.e., face the problem of distribution shift. As discussed in Section 2.4, we can use transfer learning to handle such scenarios. We will delve deep into this scenario and show how transfer learning reduces data dependence.

7 TRANSFER LEARNING

The last chapter studied a high alignment between feature spaces of two question types in VQA datasets and showed the possibility of designing future VQA datasets in a data-constrained setting. This chapter continues our study of exploiting latent embedding space to handle data constraints. We examine the problem of lexical robustness in VQA systems. We study the widely used VQA 2.0 dataset similar to the last chapters. We observe that state-of-the-art VQA models fail to perform well on rephrasings of a question, which raise important questions like *Are these models robust towards linguistic variations? Do we need to optimize architecture or the datasets?* We answer these questions by analyzing VQA models in the space of paraphrasing. We explored the role of language and cross-modal pre-training to investigate the robustness of VQA models towards lexical variations. We performed experiments that show that pre-trained language encoders generate efficient representations of question rephrasings, leading to a better understanding of VQA models. Further, we conducted an empirical study that determines the role of pre-trained language encoders in improving lexical robustness.

publication and chapter structure

The methods and results presented in this chapter have been published in Findings of EMNLP [70]. We discuss the problem definition in Section 7.1. Section 7.2 discusses the building blocks of our experiments in this study. We discuss the dataset and experimental setup in Section 7.3. Section 7.4 provides detailed result analysis of our experiments. At last, we present a discussion in Section 7.5 and summarize our chapter in Section 7.6.

7.1 PROBLEM DEFINITION

As discussed in previous chapters, Visual Question Answering is an image-conditioned question answering task. There has been significant progress in VQA since the introduction of the VQA challenge[1]. The release of new model architectures and training techniques aims to close the gap between model and oracle accuracy on benchmarking datasets like VQA 2.0 [56]. The majority of models introduce semantically rich visual features [6], multi-modal fusion techniques [49, 195], and efficient attention schemes [112, 192] to obtain higher performances on VQA.

However, during the deployment of these models into real-world scenarios, the models should be robust to linguistic variations that

1 https://visualqa.org/challenge.html

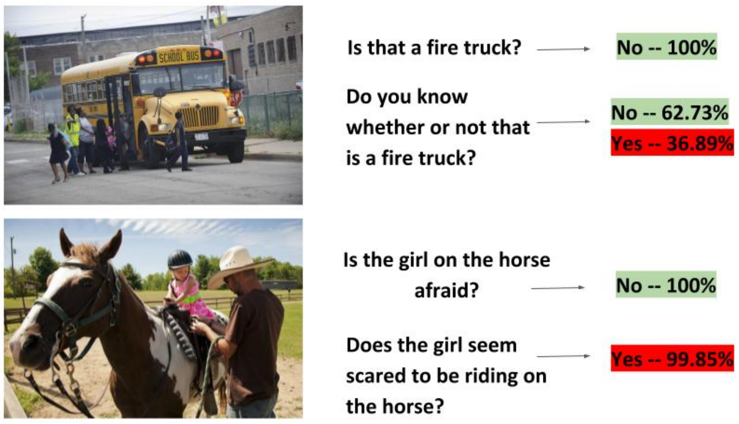

Is that a fire truck?
No -- 100%
Do you know whether or not that is a fire truck?
No -- 62.73%
Yes -- 36.89%
Is the girl on the horse afraid?
No -- 100%
Does the girl seem scared to be riding on the horse?
Yes -- 99.85%

Figure 7.1: Example from VQA-Rephrasings dataset [157]. The answers are obtained using Pythia [67] where green text refers to correct answer and red text refers to wrong answer [70].

vqa models are sensitive to lexical variations

originate from real user interactions. Recently, Shah et al. [157] showed the limitation of state-of-the-art VQA models [67, 81] to be sensitive to the lexical variations, which results in significant performance drops on test datasets when questions are replaced with their rephrasings. Figure 7.1 shows the drops in confidence scores for the VQA model when the original question is replaced with its rephrasing at test time.

Because of the limitation of current datasets that expose VQA models to a small subset of the language distribution, it disables the model's ability to generalize at test times. A plausible solution to make these systems robust to lexical variations is to collect larger datasets that account for these rephrasings. However, as explained in Chapter 1, data collection is expensive and time-consuming. Therefore, we focus on leveraging learning from pre-trained models and analyze the possibility of using them to make lexically robust VQA systems without collecting any training data. This learning comes under the umbrella of transfer learning, a method introduced in Section 2.4, that reduces the need for labeled target data by transforming models and learned representations. Our motivating question then becomes:

Upto what extend transfer learning reduce data dependence in case of data distribution shifts?

which we aim to answer in the following sections. We present an empirical study showing the role of large pre-trained Transformer-based language models in inducing similar feature embeddings for the rephrasings by strongly emphasizing on keywords. Our results establish that replacing a language encoder with a pre-trained model makes VQA models lexically robust. We show that pre-training is essential for achieving lexical robustness even with complex Transformer-based VQA architectures.

7.2 METHODS

Shah et al. [157] presented a model-agnostic cyclic consistency approach that generates question rephrasings on the fly during training to make VQA models robust to lexical variations. Their best-reported model achieves 56.59% VQA accuracy of question rephrasings. All the models used in the experiments by Shah et al. [157] incorporate an RNN-based language encoder. Although RNNs have shown great success in NLP but due to their recurrence behavior, they face the problem of long computing times and fail to learn from the long-range dependencies. Transformers overcome this problem by using attention to model dependencies between multiple words of a sequence. Transformer models [173] have led to immense improvements in the whole NLP task spectrum [178]. The core of Transformer architecture is multi-headed self-attention that encodes the relationship of every word with its neighboring words in several different representational subspaces, thus making these representations robust to linguistic variations.

previous work on making vqa robust to rephrasings

As discussed in Section 7.1, the limitation of current datasets points to collecting larger datasets that account for these linguistic variations. The expense involved in data collection led to the emergence of pre-training approaches that have obtained massive success in deep learning. Pre-trained models like ULMFiT [64], GPT [141], BERT [35] have improved performance on various NLP tasks [144, 178] for which very limited training data is available. In the space of multi-modal learning, various models have introduced ([22, 111, 170]) cross-modal pre-training methods to alleviate this problem in VQA.

leverage pre-training to handle data constraints

In this work, we study the impact of using pre-trained methods to make VQA models lexically robust. We explain the building blocks of our experiments in this study.

Sentence-BERT (SBERT) [145][2] is a BERT based language encoder that generates semantically rich embeddings. It uses siamese and triplet networks [152] to finetune BERT [35]. BERT is a pre-trained Transformer encoder trained on large amounts of monolingual data. Sentence-BERT (SBERT) obtains state-of-the-art results on common semantic textual similarity and transfer learning tasks.

BUTD by Anderson et al. [6] is a well know VQA model that uses GRU to encode input questions and uses them to attend image RoI features. It enables region-based attention to generate an answer. Many other VQA architectures like Pythia [67], and BAN [81] have BUTD as their base architecture.

2 https://github.com/UKPLab/sentence-transformers

LXMERT by Tan and Bansal [170] is a vision-language cross-modality pre-training network. In contrast to single modality pre-training in BERT, LXMERT focuses on vision-language pre-training. This multi-modal pre-training helps to understand better visual contents, language semantics, and their relationships. There are three Transformer encoders: an object relationship encoder, a language encoder, and a cross-modality encoder. They are pre-trained using five different vision-language tasks[3] We used LXMERT as a placeholder for Transformer based VQA architectures to investigate if a model architecture plays any role in improving the robustness of lexical variations.

7.3 EXPERIMENTAL SETUP

7.3.1 *Dataset*

We used the training split of the VQA 2.0 dataset to train our models and evaluate the performance against the VQA-Rephrasings (VQA-R) dataset. VQA-R by Shah et al. [157] contains 40,504 image-question pairs randomly sampled from the validation split of VQA 2.0. For each question, Shah et al. [157] collected three rephrasings using human annotators, which amount to 121,512 pairs. The authors ensured that rephrasings were syntactically correct and semantically aligned with the original questions during data collection. We call the original split ORI in our experiments, and rephrasings split as REP.

7.3.2 *Implementation Details*

We used the BUTD architecture and trained it using Adamax [82]. Similar to experiments in previous chapters, we used 36 RoI per image to obtain visual features. We trained the model using entire training data with an initial learning rate of 2 x 10^{-3}. We used ReLU activation units and reported the standard validation accuracy [7] for each split of the VQA-Rephrasings dataset. Since BUTD contains a GRU for encoding questions, we replaced GRU with SBERT to use a pre-trained Transformer-based language encoder for questions. We call this BUTD+SBERT in our experiments. The question embeddings from SBERT are passed through a fully-connected (FC) layer, which is then combined with image embeddings to produce a multi-modal representation of the question-image pair. The size of SBERT embeddings and FC layer is 768 and 512.

For LXMERT, we trained three variants. For the first variant, we randomly initialized all parameters. We initialized only the language encoder with BERT weights for our second variant. Finally, we initialize all parameters except the VQA task head with the pre-trained LXMERT

3 Please refer to Section Section 2.3.2 of Chapter 2 for more details.

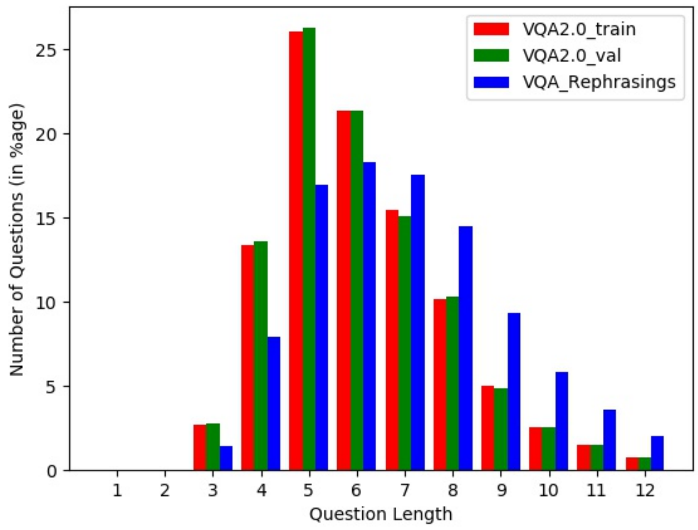


Figure 7.2: Dataset statistics about the number of questions (in percentage) with varying lengths for three subsets of VQA namely training and validation data of VQA2.0, and VQA-Rephrasings.

weights[4] for the third variant. We perform these experiments to see the impact of pre-training at different stages towards making lexical robust VQA systems. For the fairness of results on VQA-R, we don't use any part of the validation split of VQA 2.0 during training or finetuning. We use the default hyperparameters set as in the original implementation. The first, second, and third variants converged at 17 (30 hours), 10 (18 hours), and 4 epochs (8 hours), respectively, on Nvidia V100 GPU.

7.4 RESULT ANALYSIS

7.4.1 *Syntactic Variation causes Data Distribution Shift*

Machine learning models generally perform well when training and testing distributions are similar. In the case of test samples drawn from a different distribution than training, ML models suffer from generalization issues, i.e., fail to perform well at test times. However, Agrawal, Batra, and Parikh [2] and Wang et al. [180] showed the limitation of these models to be misled by contextual heuristics in the training data instead of learning underlying generalizations. A similar trend in NLI was also observed where McCoy, Pavlick, and Linzen [118] found that SOTA language models like BERT also adopted underlying heuristics, thus failing to generalize for test samples. Figure 7.2 shows the distribution of question lengths of VQA 2.0-train, VQA 2.0-

generalization problems when training and testing distributions differ

4 https://github.com/airsplay/lxmert

Model	VQA-Rephrasings									
	ORI					REP				
	OA	NUM	Y/N	O	RG	OA	NUM	Y/N	O	RG
BUTD	63.13	41.53	81.27	54.98	-	54.27	33.08	75.73	43.52	-
BUTD+SBERT	62.50	40.22	81.46	53.91	-0.99	57.21	35.91	77.46	47.40	+5.42
LXMERT (a)	63.86	43.38	81.86	55.54	-	54.79	33.86	75.73	44.36	-
LXMERT (b)	64.86	44.32	83.22	56.28	+1.56	58.21	39.25	78.8	47.55	+6.24
LXMERT (c)	73.61	55.88	88.56	66.9	+15.26	66.27	50.63	83.32	57.42	+20.95

Table 7.1: VQA Accuracy results on both splits of VQA-R. OA refers to overall accuracy. NUM, Y/N and O refers to accuracies for number, yes/no and other answer class. RG refers to relative gain. RG for BUTD+SBERT and LXMERT (c) (and LXMERT (b)) are computed w.r.t BUTD and LXMERT (a) respectively.

val, and VQA-R. We observe that the training and validation split of VQA 2.0 have similar distributions, while the distribution of VQA-R is different. Table 7.1 shows that BUTD performs better on ORI than REP for VQA-R. It is because the language encoder of BUTD that uses VQA 2.0-train for training has a similar distribution as that of ORI. Therefore, a shift in the lexical distribution of REP contributes to this artifact.

7.4.2 *Pre-trained Language Encoders generate Lexically Robust Representations*

The two splits of VQA-R, REP, and ORI, contain the same semantic information. Due to the poor representation of input questions by GRU, we observe significant performance drops for REP. One can alleviate this problem by using a better language encoder. Therefore, considering the robustness of SBERT to lexical variations, we replace the GRU with SBERT in BUTD. SBERT will efficiently extract the overall semantics. Table 7.1 shows the results of our approach BUTD+SBERT, where relative to BUTD, our approach improves the accuracy of REP by 5.41%. It also performs slightly better than BAN+CC, which is the reported SOTA model of Shah et al. [157]. It is worth noting that BUTD has a relatively simpler architecture than BAN, and we don't train any auxiliary component like the question generation module in CC. However, for ORI, whose distribution is similar to VQA 2.0-train, BUTD+SBERT obtains comparable performance. Compared to generalized embeddings from SBERT, which never interact with VQA language data, GRU is trained on the training data of VQA 2.0, which generates semantically rich question embeddings for ORI. A similar trend was observed by Tan and Bansal [170] in VQA 2.0-dev accuracies when they used BERT as a language encoder. Considering that

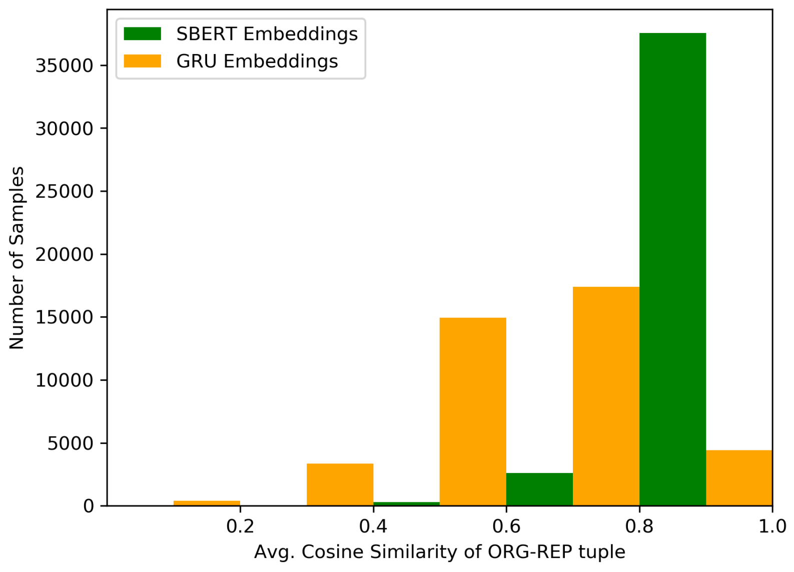


Figure 7.3: Distribution of cosine similarity of ORG-REP tuples, where each tuple comprises of 1 original sentence and its 3 rephrasings. We calculate the average cosine similarity of rephrasings with its original sentence.

SBERT doesn't directly improve VQA models, we ask ourselves *What are the underlying factors that allow SBERT to improve the REP accuracy?*

pre-trained language encoder is robust to lexical variations

We answer the above question by generating the SBERT & GRU embeddings for the original question and its three rephrases. We calculate the average cosine similarities of the paraphrases with their original counterpart. Figure 7.3 shows distribution of cosine similarity. We observe that SBERT moves the embeddings of rephrases in its representational vector space significantly closer to the original question. On the other hand, due to its lexical sensitivity, GRU fails to extract the underlying common semantics. The average cosine similarity of ORG-REP tuple for SBERT and GRU is 91% and 60%, respectively. Hence, we conclude that the pre-trained language encoder drives the major accuracy gains for REP, making our approach model-agnostic.

7.4.3 *Pre-trained Language Encoders latch on Keywords*

Keywords play an essential role in a sentence as they carry fundamental semantics. Since rephrases or paraphrases are semantically similar sentences with lexical variations, they share common keywords to control their semantics. A lexically robust encoder must latch on these keywords to generate semantically rich vector representations. We build an ordered sequence of keywords S1 extracted from a complete sentence S2 in our experiments. We use rake-nltk to extract keywords. We use a language encoder to encode both S1 and S2 and measure the cosine similarity of the pair. We hypothesize that a lexi-

pre-trained language encoders latch on keywords

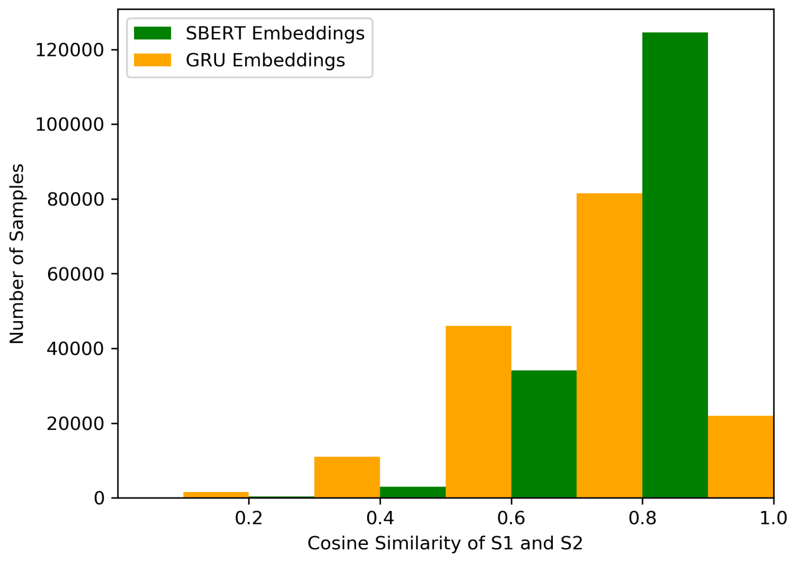


Figure 7.4: Distribution of cosine similarity of sentence S1 and S2. S1 is a question from VQA-Rephrasing dataset and S2 is an ordered sequence of keywords obtained from S1.

cally robust language encoder generates similar representations of S1 and S2 in a vector space. The average cosine similarity over the whole VQA-R dataset for SBERT and GRU is 0.85 and 0.64, respectively. Figure 7.4 shows the distribution of cosine similarity of S1 and S2 over whole VQA-R. SBERT's ability to stress on keywords allows it to circumvent syntactic deviations in paraphrases and embed them closer in the vector space.

7.4.4 *Transformers are Good but Pre-training makes them Great*

In contrast to single modal pre-training, LXMERT is conditioned on both vision & language modality; therefore, it generates better multi-modal representations. It is justified by the results in Table 7.1 where LXMERT (c) achieves SOTA results on both ORI and REP. Since multiple questions are associated with a single image, cross-modal attention helps obtain efficient language representations, making VQA models robust toward question rephrasings. However, the high performance of LXMERT (c) raises an important question *Are the gains coming from pre-training or LXMERT architecture?*

cross-modal pre-training achieves SOTA results on rep split

We answer this question with our first variant, LXMERT (a), where LXMERT (a) achieves similar performance to BUTD on REP split. It shows that even a complex cross-modal architecture is insufficient for designing lexically robust VQA systems. Nevertheless, in our second version, where we train LXMERT initialized with BERT weights, we observe relative gains of 1.56% in ORI and 6.24% in REP. At last, fine-tuning LXMERT with pre-trained language, vision, and cross-modality encoders boosts the performance in REP to 20.95% relative to LXMERT (a).

In contrast to single modal pre-training like BERT, that only captures intra-modal relationships, VL pre-training like LXMERT-(c) learns cross-modality relationships. Cross-modal attention aligns entities across input modalities. Therefore, it induces semantically rich and robust joint representations, outperforming BERT only initialization. These results validate that pre-training is crucial for obtaining lexical robustness, even for highly complex architectures.

7.5 DISCUSSION

Pre-trained language models like BERT are trained on large and diverse datasets. Therefore, it is generally hypothesized that such models are robust to lexical variations. Our results show that pre-trained language encoders like SBERT improve the performance of REP split by 5.42% relative to a GRU encoder; it still underperforms by 9.37% relative to ORI questions which are semantically similar. A similar trend was observed for task-specific multimodal pre-training. We observe that LXMERT (c) struggles to close the relative performance gap of about 10% between ORI and REP. We leave this study as future work and show the effectiveness of pre-training in making models lexically robust in this study.

7.6 SUMMARY

We started this chapter with our motivating question:

Upto what extend transfer learning reduce data dependence in case of data distribution shifts?

We studied the problem of making VQA models robust to lexical variations. We analyze the newly released VQA-Rephrasings dataset containing two splits, namely ORI and REP. The distribution shifts between ORI and REP lead to poor performance on REP which includes rephrasings of questions in the ORI split. We showed the role of single modal and multi-modal pre-training in making lexically robust VQA models. We conducted experiments using SBERT and BUTD model and showed that pre-trained language encoders produce semantically similar embeddings for multiple rephrases of a sentence by latching on keywords. They make VQA models robust to lexical variations. We presented an extensive study of training three variants of LXMERT, a Transformer-based model, and showed that pre-training is crucial for obtaining lexical robustness, even for highly complex architectures. At last, we obtain SOTA results on the VQA-Rephrasings dataset using cross-modal pre-training with Transformer-based VQA architectures.

Part V

CONCLUSION AND FUTURE WORK

8 CONCLUSION

The rise of DL has embraced the development of many real-world applications that aim to steer the progress of the human race and make their life easy. Among multiple applications like translation systems and summarization tools, personal digital assistants are a successful application where users rely on these assistants to carry out various daily tasks. NLP tasks like NLG and NLU are core components for building these personal assistants, and their advancements directly translate into accuracy improvements. However, the NLU and NLG systems are trained using tremendous amounts of training examples like any other DL model. These training examples enable a model's learning and allow them to generalize well at test times. Nevertheless, we need to employ human annotators for data collection, making this process expensive in terms of time, money, and effort. This strong dependence of DL applications on data collection seems to be a bottleneck, thus, limiting its use to mere big corporations. Therefore, we began this thesis with the motivating question:

we need training datasets for designing better nlp systems

collection of training data is an expensive process

> Can we develop general methods to build data-constrained NLP systems?

In other words,

> Can we develop NLU and NLG systems with limited labeled data?

A data-constrained setting is when we have minimal training examples. As mentioned in Section 2.4, some standard methods, e.g., data augmentation, minimal supervision, data selection and transfer learning are used to address a data-constrained setting. In this thesis, we investigated the role of these methods in overcoming performance drops emerging from limited training examples.

For the first part, we presented a **data augmentation** (Chapter 3) approach. Data augmentation is the most straightforward approach when only limited parallel examples are available. We studied a scenario of adding a new feature for a task-oriented dialogue agent, for which we have minimal training data. The limited training data degrades the agent's performance on intent classification and slot labeling downstream tasks. Avoiding the expensive way of data collection for the problem handling, we proposed the *interpretation-to-text* paraphrase generation model. Our model is independent of any parallel data and uses existing training data for bootstrapping new features.

generating high-quality synthetic data from limited parallel data

This way, we improved the accuracy of downstream tasks in a data-constrained setting. We showed our approach's usefulness by experimenting on a public dataset in English and a commercial dialogue system with real-life data in the German language.

improving distant supervision with unsupervised nlg algorithm to minimize semantic information loss

Following our study of limited training data, we explored few-shot data-to-text generation in Chapter 4. We observed that tabular data summarization faces a *low semantic coverage* problem in a few-shot setting. In other words, important table values are absent in the generated sentence. To overcome this, we use a small unlabeled dataset and employ **distant supervision** to generate sentence outputs for these unlabeled tables and use these samples for training. However, the *low semantic coverage* problem persists. Therefore, we proposed an unsupervised algorithm that exploits pre-trained language models to fill the missing slots and improve semantic coverage. Our algorithm, also called search-and-learn, employs a two-stage finetuned model that learns from search results to produce fluent text and improves inference efficiency significantly. We tested our approach on two open-source data-to-text datasets. We showed the effectiveness of our approach, which recovers a majority of missing input table slots on these datasets. Primarily, we cover 98.35% of input slots on E2E, largely alleviating the *low coverage problem* and closing the gap between few-shot and fully supervised learning.

use already available inter-annotator agreement to choose the smallest and most informative training subset

The above approaches use entire training data to generate more examples or perform distant supervision. One can use whole training data when the number of examples is scarce. But, we need to pick the most informative examples while many examples are available. This selection of the most informative subset of training data will allow us to train the models with minimum compute resources and considerably reduce the training times. We studied the known problem of VQA systems and present a **data selection** method for VQA systems (Chapter 5). We designed a diagnostic tool called EaSe that selects the most informative training examples by quantifying the difficulty of an `image, question` example. Our tool leveraged the pattern of answers given by multiple human annotators for a given question. In particular, EaSe considers two aspects of the answers: their entropy and their semantic content. We used the readily available information in any VQA dataset for computing EaSe scores. Our experiments on two open-source datasets showed the effectiveness of our diagnostic, where we recovered a significant portion of model accuracy by using the smallest training subset.

The above approaches were applicable only for generating and extending task-specific datasets. However, in real-world scenarios, one can encounter distribution shifts where all the above methods don't apply. To handle such systems, we investigated the modeling part and showed how exploiting latent representations could control data constraints. We studied the over-representation of polar questions in VQA datasets and its relationship with non-polar questions for the widely used VQA 2.0 dataset in Chapter 6. We analyzed the feature embedding of these two question types. We performed a cross-polarity evaluation to examine the alignment of polar and non-polar feature spaces. Our results of exploring the feature space of visual-text embedding show that we could use polar questions to answer non-polar questions; if polar questions used for training refer to the semantic concepts being considered in the non-polar questions. Therefore, one can easily use available polar questions to augment future VQA datasets to enable better systems in a data-constrained setting.

exploiting feature embedding space to handle data constraints

Following our study of exploiting latent representations, we investigated the use of **transfer learning** to make VQA models lexically robust (Chapter 7). Like the last chapters, we analyzed the widely used VQA 2.0 dataset and observed the limitation of state-of-the-art VQA models to perform well on question rephrasings at test times. We avoided the plausible solution of collecting larger datasets that account for these rephrasings. We showed that large pre-trained Transformer based language models induce similar feature embeddings for the rephrasings by strongly emphasizing on keywords. Our results showed that replacing the language encoder with a pre-trained model makes VQA models lexically robust. We achieved SOTA results on the VQA-Rephrasings dataset by combining crossmodal pre-training with Transformer based VQA architectures.

transfer learning to handle data-constraints

To summarize, we have achieved the following goals we set in Chapter 1 to answer our research question.

- **Methods for automatic data generation and selection:** We showed the effectiveness of using deep learning models for generating high-quality synthetic data for bootstrapping new features with limited training data. We presented an unsupervised algorithm that minimizes semantic information loss in a few-shot setting for data-to-text systems. In addition to data generation methods, we also proposed a data selection tool that selects the smallest and most informative training examples.
- **Exploiting latent embedding space to handle data constraints:** We showed high alignment between polar and non-polar questions for VQA datasets and presented the possibility of designing future VQA datasets in a data-constrained setting. We showed the effectiveness of replacing the language encoder with a pre-trained model to make lexically robust VQA systems bypassing the data collection requirement.

9 FUTURE WORK

This chapter concludes our work by listing some promising future directions. Our data generation/selection methods and exploiting latent embedding space have shown promising results in designing NLG and NLU systems in a data-constrained setting. However, there are some future directions one can take for further improvements.

Our data augmentation approach showed improvements in the downstream tasks with minimal seed training data. However, finding paraphrases that hold properties of being novel, diverse, and semantically preserved is challenging due to two different models with hyper-parameters and the sampling parameters. Finding an efficient way of choosing these parameters will benefit the model. Furthermore, future experiments should study the optimal number of paraphrases that needs to be added for a new feature and what is the performance of slot shuffling across various domains.

efficient hyper-parameter tuning

We use slot recombination for generating new unlabeled tables in our few-shot data-to-text generation. We observed that the recombining table slots did not give a good performance as compared to the original unlabeled tables. We hypothesize that since new tables introduce more slot values, it is helpful for few-shot data-to-text generation, where only a few hundred parallel examples are available. However, further experiments need to be designed to find better recombinations that could overcome this limitation.

better slot recombination

In our study of data selection, our proposed tool EaSe studies visual question answering. It quantifies the difficulty of a VQA example based on its pattern of answers. We recovered a significant portion of accuracy after training VQA models on the smallest and most informative subset selected by EaSe. Further experiments need to be designed to combine model prediction for difficulty estimation in EaSe. We would also like to apply EaSe on other NLP applications where more than one human annotations are available.

combine model prediction

We analyzed the influence of polar and non-polar questions on each other while using them jointly to train VQA systems. Our results indicated we could solve the problem of VQA for non-polar questions using polar questions, as long as polar questions cover relevant non-polar topics. We intend to study the empirical extent to which this phenomenon holds. Our study highlighted the possibility of designing VQA datasets in cost-efficient ways by collecting more polar questions. We want to explore automated methods using NLP tools to convert non-polar questions into polar ones. We are interested in measuring how we can use polar examples to model several non-polar con-

automatic polar question generation

cepts in a joint visual-text space. Therefore, we can explicitly impose an arbitrary number of concepts in the joint feature space while keeping a fixed 2-dimensional classification objective. The training regime resembles the conditions of Generative Adversarial Networks (GAN). It could open the possibility of learning new classes over time, which has potential applications in continuous learning.

close gaps between ORI and REP

At last, to make VQA models robust to lexical variations, we explored the role of pre-trained language encoders. We showed that encoders like SBERT produce semantically similar embeddings for multiple rephrasings by latching on keywords. We observe SOTA results on the VQA-Rephrasings dataset. However, despite extensive cross-modal pretraining, the gap between ORI and REP is not reduced. It would be interesting to investigate factors that prevent closing the accuracy gap. Furthermore, future experiments can study *reasons behind more accuracy gains for some classes like numbers than the others.*

BIBLIOGRAPHY

[1] Oliver Adams, Adam Makarucha, Graham Neubig, Steven Bird, and Trevor Cohn. "Cross-Lingual Word Embeddings for Low-Resource Language Modeling." In: *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*. Valencia, Spain: Association for Computational Linguistics, Apr. 2017, pp. 937–947. URL: https://aclanthology.org/E17-1088.

[2] Aishwarya Agrawal, Dhruv Batra, and Devi Parikh. "Analyzing the Behavior of Visual Question Answering Models." In: *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing* (2016). DOI: 10.18653/v1/d16-1203. URL: http://dx.doi.org/10.18653/v1/D16-1203.

[3] Aishwarya Agrawal, Dhruv Batra, Devi Parikh, and Aniruddha Kembhavi. "Don't just assume; look and answer: Overcoming priors for visual question answering." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018.

[4] Aishwarya Agrawal, Aniruddha Kembhavi, Dhruv Batra, and Devi Parikh. "C-vqa: A compositional split of the visual question answering (vqa) v1. 0 dataset." In: *arXiv preprint arXiv:1704.08243* (2017).

[5] Md Zahangir Alom, Tarek M Taha, Christopher Yakopcic, Stefan Westberg, Paheding Sidike, Mst Shamima Nasrin, Brian C Van Esesn, Abdul A S Awwal, and Vijayan K Asari. "The history began from alexnet: A comprehensive survey on deep learning approaches." In: *arXiv preprint arXiv:1803.01164* (2018).

[6] Peter Anderson, Xiaodong He, Chris Buehler, Damien Teney, Mark Johnson, Stephen Gould, and Lei Zhang. "Bottom-up and top-down attention for image captioning and visual question answering." In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018, pp. 6077–6086.

[7] Stanislaw Antol, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, C Lawrence Zitnick, and Devi Parikh. "Vqa: Visual question answering." In: *Proceedings of the IEEE international conference on computer vision*. 2015, pp. 2425–2433.

[8] Pepa Atanasova, Jakob Grue Simonsen, Christina Lioma, and Isabelle Augenstein. "Generating Fact Checking Explanations." In: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. 2020, pp. 7352–7364.

[9] Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary Ives. "Dbpedia: A nucleus for a web of open data." In: *The semantic web*. Springer, 2007, pp. 722–735.

[10] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. "Layer normalization." In: *arXiv preprint arXiv:1607.06450* (2016).

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

[12] Ramakrishna Bairi, Rishabh Iyer, Ganesh Ramakrishnan, and Jeff Bilmes. "Summarization of multi-document topic hierarchies using submodular mixtures." In: *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. 2015, pp. 553–563.

[13] Emily Bender. "The BenderRule: On Naming the Languages We Study and Why It Matters." In: *The Gradient* (2019).

[14] Yoshua Bengio, Réjean Ducharme, and Pascal Vincent. "A neural probabilistic language model." In: *Advances in Neural Information Processing Systems* 13 (2000).

[15] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. "Language models are few-shot learners." In: *Advances in neural information processing systems* 33 (2020), pp. 1877–1901.

[16] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. "End-to-end object detection with transformers." In: *European conference on computer vision*. Springer. 2020, pp. 213–229.

[17] Alessandra Cervone, Chandra Khatri, Rahul Goel, Behnam Hedayatnia, Anu Venkatesh, Dilek Hakkani-Tur, and Raefer Gabriel. "Natural Language Generation at Scale: A Case Study for Open Domain Question Answering." In: *Proceedings of the 12th International Conference on Natural Language Generation*. 2019, pp. 453–462.

[18] Wei-Lun Chao, Hexiang Hu, and Fei Sha. "Being Negative but Constructively: Lessons Learnt from Creating Better Visual Question Answering Datasets." In: *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*. 2018, pp. 431–441.

[19] Hongshen Chen, Xiaorui Liu, Dawei Yin, and Jiliang Tang. "A survey on dialogue systems: Recent advances and new frontiers." In: *Acm Sigkdd Explorations Newsletter* 19.2 (2017), pp. 25–35.

[20] Wenhu Chen, Jianshu Chen, Yu Su, Zhiyu Chen, and William Yang Wang. "Logical natural language generation from open-domain tables." In: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. July 2020, pp. 7929–7942. DOI: 10.18653/v1/2020.acl-main.708. URL: https://www.aclweb.org/anthology/2020.acl-main.708.

[21] Xie Chen, Yu Wu, Zhenghao Wang, Shujie Liu, and Jinyu Li. "Developing real-time streaming transformer transducer for speech recognition on large-scale dataset." In: *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE. 2021, pp. 5904–5908.

[22] Yen-Chun Chen, Linjie Li, Licheng Yu, Ahmed El Kholy, Faisal Ahmed, Zhe Gan, Yu Cheng, and Jingjing Liu. "UNITER: Universal image-text representation learning." In: *European Conference on Computer Vision*. Springer. 2020, pp. 104–120.

[23] Zhiyu Chen, Harini Eavani, Wenhu Chen, Yinyin Liu, and William Yang Wang. "Few-Shot NLG with Pre-Trained Language Model." In: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. 2020, pp. 183–190.

[24] Zhiyu Chen, Harini Eavani, Wenhu Chen, Yinyin Liu, and William Yang Wang. "Few-shot NLG with pre-trained language model." In: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. July 2020, pp. 183–190. DOI: 10.18653/v1/2020.acl-main.18. URL: https://www.aclweb.org/anthology/2020.acl-main.18.

[25] Kashyap Chitta, José M Álvarez, Elmar Haussmann, and Clément Farabet. "Training data subset search with ensemble active learning." In: *IEEE Transactions on Intelligent Transportation Systems* (2021).

[26] Eunah Cho, He Xie, and William M. Campbell. "Paraphrase Generation for Semi-Supervised Learning in NLU." In: *Proceedings of the Workshop on Methods for Optimizing and Evaluating NLG*. Minneapolis, Minnesota, 2019, pp. 45–54. DOI: 10.18653/v1/W19-2306. URL: https://www.aclweb.org/anthology/W19-2306.

[27] 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: *arXiv preprint arXiv:1406.1078* (2014).

[28] James Clarke and Mirella Lapata. "Discourse constraints for document compression." In: *Computational Linguistics* 36.3 (2010), pp. 411–441.

[29] Ronan Collobert and Jason Weston. "A unified architecture for natural language processing: Deep neural networks with multitask learning." In: *Proceedings of the 25th international conference on Machine learning*. 2008, pp. 160–167.

[30] Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. "Natural language processing (almost) from scratch." In: *Journal of machine learning research* 12.ARTICLE (2011), pp. 2493–2537.

[31] Corinna Cortes and Vladimir Vapnik. "Support-vector networks." In: *Machine learning* 20.3 (1995), pp. 273–297.

[32] Alice Coucke, Alaa Saade, Adrien Ball, Théodore Bluche, Alexandre Caulier, David Leroy, Clément Doumouro, Thibault Gisselbrecht, Francesco Caltagirone, Thibaut Lavril, et al. "Snips voice platform: an embedded spoken language understanding system for private-by-design voice interfaces." In: *arXiv preprint arXiv:1805.10190* (2018).

[33] David R Cox. "The regression analysis of binary sequences." In: *Journal of the Royal Statistical Society: Series B (Methodological)* 20.2 (1958), pp. 215–232.

[34] Xiang Dai and Heike Adel. "An Analysis of Simple Data Augmentation for Named Entity Recognition." In: *Proceedings of the 28th International Conference on Computational Linguistics*. Barcelona, Spain (Online): International Committee on Computational Linguistics, Dec. 2020, pp. 3861–3867. DOI: 10.18653/v1/2020.coling-main.343. URL: https://aclanthology.org/2020.coling-main.343.

[35] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. "Bert: Pre-training of deep bidirectional transformers for language understanding." In: *arXiv preprint arXiv:1810.04805* (2018).

[36] Bhuwan Dhingra, Manaal Faruqui, Ankur Parikh, Ming-Wei Chang, Dipanjan Das, and William Cohen. "Handling divergent reference texts when evaluating table-to-text generation." In: *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. 2019, pp. 4884–4895. URL: https://www.aclweb.org/anthology/P19-1483.

[37] Quynh Do and Judith Gaspers. "Cross-lingual Transfer Learning with Data Selection for Large-Scale Spoken Language Understanding." In: *Proceedings of the 2019 Conference on EMNLP and the 9th IJCNLP*. Hong Kong, China, 2019, pp. 1455–1460. DOI: 10.18653/v1/D19-1153. URL: https://www.aclweb.org/anthology/D19-1153.

[38] George Doddington. “Automatic evaluation of machine translation quality using n-gram co-occurrence statistics.” In: *Proceedings of the Second International Conference on Human Language Technology Research*. 2002, pp. 138–145. URL: https://dl.acm.org/doi/10.5555/1289189.1289273.

[39] Linhao Dong, Shuang Xu, and Bo Xu. “Speech-transformer: a no-recurrence sequence-to-sequence model for speech recognition.” In: *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE. 2018, pp. 5884–5888.

[40] Yue Dong, Zichao Li, Mehdi Rezagholizadeh, and Jackie Chi Kit Cheung. “EditNTS: An neural programmer-interpreter model for sentence simplification through explicit editing.” In: *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. 2019, pp. 3393–3402. URL: https://www.aclweb.org/anthology/P19-1331.

[41] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. “An image is worth 16x16 words: Transformers for image recognition at scale.” In: *arXiv preprint arXiv:2010.11929* (2020).

[42] Jiaju Du, Fanchao Qi, and Maosong Sun. “Using BERT for word sense disambiguation.” In: *arXiv preprint arXiv:1909.08358* (2019).

[43] S Durga, Rishabh Iyer, Ganesh Ramakrishnan, and Abir De. “Training Data Subset Selection for Regression with Controlled Generalization Error.” In: *International Conference on Machine Learning*. PMLR. 2021, pp. 9202–9212.

[44] Ondřej Dušek, David M. Howcroft, and Verena Rieser. “Semantic noise matters for neural natural language generation.” In: *Proceedings of the 12th International Conference on Natural Language Generation*. 2019, pp. 421–426. URL: https://aclanthology.org/W19-8652.

[45] Ondřej Dušek, Jekaterina Novikova, and Verena Rieser. “Evaluating the state-of-the-art of End-to-End Natural Language Generation: The E2E NLG challenge.” In: *Computer Speech & Language* 59 (2020), pp. 123–156. ISSN: 0885-2308. DOI: https://doi.org/10.1016/j.csl.2019.06.009.

[46] Jeffrey L Elman. “Finding structure in time.” In: *Cognitive science* 14.2 (1990), pp. 179–211.

[47] Roger Evans, Paul Piwek, and Lynne Cahill. “What is NLG?” In: *Proceedings of the International Natural Language Generation Conference*. 2002, pp. 144–151.

[48] Markus Freitag and Scott Roy. "Unsupervised Natural Language Generation with Denoising Autoencoders." In: *Proceedings of the 2018 Conference on EMNLP*. Brussels, Belgium, 2018, pp. 3922–3929. DOI: 10.18653/v1/D18-1426. URL: https://www.aclweb.org/anthology/D18-1426.

[49] Akira Fukui, Dong Huk Park, Daylen Yang, Anna Rohrbach, Trevor Darrell, and Marcus Rohrbach. "Multimodal Compact Bilinear Pooling for Visual Question Answering and Visual Grounding." In: *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing* (2016). DOI: 10.18653/v1/d16-1044. URL: http://dx.doi.org/10.18653/v1/D16-1044.

[50] Hagen Fürstenau and Mirella Lapata. "Semi-Supervised Semantic Role Labeling." In: *Proceedings of the 12th Conference of the EACL*. Athens, Greece, 2009, pp. 220–228. URL: https://www.aclweb.org/anthology/E09-1026.

[51] Claire Gardent, Anastasia Shimorina, Shashi Narayan, and Laura Perez-Beltrachini. "Creating training corpora for nlg micro-planning." In: *55th annual meeting of the Association for Computational Linguistics (ACL)*. 2017.

[52] Judith Gaspers, Penny Karanasou, and Rajen Chatterjee. "Selecting Machine-Translated Data for Quick Bootstrapping of a Natural Language Understanding System." In: *Proceedings of the 2018 Conference of the NAACL-HLT*. New Orleans, USA, 2018, pp. 137–144. DOI: 10.18653/v1/N18-3017. URL: https://www.aclweb.org/anthology/N18-3017.

[53] Efstratios Gavves, Thomas Mensink, Tatiana Tommasi, Cees GM Snoek, and Tinne Tuytelaars. "Active transfer learning with zero-shot priors: Reusing past datasets for future tasks." In: *Proceedings of the IEEE International Conference on Computer Vision*. 2015, pp. 2731–2739.

[54] Heng Gong, Yawei Sun, Xiaocheng Feng, Bing Qin, Wei Bi, Xiaojiang Liu, and Ting Liu. "TableGPT: Few-shot table-to-text generation with table structure reconstruction and content matching." In: *Proceedings of the 28th International Conference on Computational Linguistics*. 2020, pp. 1978–1988. URL: https://www.aclweb.org/anthology/2020.coling-main.179.pdf.

[55] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016.

[56] Yash Goyal, Tejas Khot, Douglas Summers-Stay, Dhruv Batra, and Devi Parikh. "Making the v in vqa matter: Elevating the role of image understanding in visual question answering." In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 6904–6913.

[57] Ankush Gupta, Arvind Agarwal, Prawaan Singh, and Piyush Rai. “A Deep Generative Framework for Paraphrase Generation.” In: *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence*. 2018, pp. 5149–5156.

[58] Danna Gurari and Kristen Grauman. “CrowdVerge: Predicting if people will agree on the answer to a visual question.” In: *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*. 2017, pp. 3511–3522.

[59] Danna Gurari, Qing Li, Abigale J Stangl, Anhong Guo, Chi Lin, Kristen Grauman, Jiebo Luo, and Jeffrey P Bigham. “Vizwiz grand challenge: Answering visual questions from blind people.” In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 3608–3617.

[60] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. “Deep residual learning for image recognition.” In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 770–778.

[61] Michael A. Hedderich, Lukas Lange, Heike Adel, Jannik Strötgen, and Dietrich Klakow. “A Survey on Recent Approaches for Natural Language Processing in Low-Resource Scenarios.” In: *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Online: Association for Computational Linguistics, June 2021, pp. 2545–2568. DOI: `10.18653/v1/2021.naacl-main.201`. URL: `https://aclanthology.org/2021.naacl-main.201`.

[62] Sepp Hochreiter. “Untersuchungen zu dynamischen neuronalen Netzen.” In: *Diploma, Technische Universität München* 91.1 (1991).

[63] Sepp Hochreiter and Jürgen Schmidhuber. “Long short-term memory.” In: *Neural computation* 9.8 (1997), pp. 1735–1780.

[64] Jeremy Howard and Sebastian Ruder. “Universal language model fine-tuning for text classification.” In: *arXiv preprint arXiv:1801.06146* (2018).

[65] Baotian Hu, Zhaopeng Tu, Zhengdong Lu, Hang Li, and Qingcai Chen. “Context-Dependent Translation Selection Using Convolutional Neural Network.” In: *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*. 2015, pp. 536–541.

[66] Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, and Trevor Darrell. “Caffe: Convolutional architecture for fast feature embedding.” In: *Proceedings of the 22nd ACM international conference on Multimedia*. 2014, pp. 675–678.

[67] Yu Jiang, Vivek Natarajan, Xinlei Chen, Marcus Rohrbach, Dhruv Batra, and Devi Parikh. "Pythia v0. 1: the winning entry to the vqa challenge 2018." In: *arXiv preprint arXiv:1807.09956* (2018).

[68] Justin Johnson, Bharath Hariharan, Laurens van der Maaten, Li Fei-Fei, C Lawrence Zitnick, and Ross Girshick. "Clevr: A diagnostic dataset for compositional language and elementary visual reasoning." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2017.

[69] Shailza Jolly, Tobias Falke, Caglar Tirkaz, and Daniil Sorokin. "Data-efficient paraphrase generation to bootstrap intent classification and slot labeling for new features in task-oriented dialog systems." In: *Proceedings of the 28th International Conference on Computational Linguistics: Industry Track*. 2020, pp. 10–20.

[70] Shailza Jolly and Shubham Kapoor. "Can Pre-training help VQA with Lexical Variations?" In: *Findings of the Association for Computational Linguistics: EMNLP 2020*. 2020, pp. 2863–2868.

[71] Shailza Jolly, Sebastian Palacio, Joachim Folz, Federico Raue, Jorn Hees, and Andreas Dengel. "P≈NP, at least in Visual Question Answering." In: *2020 25th International Conference on Pattern Recognition (ICPR)*. IEEE Computer Society. 2021, pp. 2748–2754.

[72] Shailza Jolly, Sandro Pezzelle, Tassilo Klein, Andreas Dengel, and Moin Nabi. "The wisdom of masses: majority, subjectivity, and semantic similarity in the evaluation of VQA." In: *arXiv preprint arXiv:1809.04344* (2018).

[73] Shailza Jolly, Sandro Pezzelle, and Moin Nabi. "EaSe: A Diagnostic Tool for VQA Based on Answer Diversity." In: *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 2021, pp. 2407–2414.

[74] Shailza Jolly, Zi Xuan Zhang, Andreas Dengel, and Lili Mou. "Search and Learn: Improving Semantic Coverage for Data-to-Text Generation." In: *Proceedings of the 36th AAAI Conference on Artificial Intelligence*. 2022.

[75] Juraj Juraska, Panagiotis Karagiannis, Kevin Bowden, and Marilyn Walker. "A deep ensemble model with slot alignment for sequence-to-sequence natural language generation." In: *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*. June 2018, pp. 152–162. DOI: 10.18653/v1/N18-1014. URL: https://www.aclweb.org/anthology/N18-1014.

[76] Kushal Kafle and Christopher Kanan. “An analysis of visual question answering algorithms.” In: *Computer Vision (ICCV), 2017 IEEE International Conference on*. IEEE. 2017, pp. 1983–1991.

[77] Kushal Kafle and Christopher Kanan. “An analysis of visual question answering algorithms.” In: *Proceedings of the IEEE International Conference on Computer Vision*. 2017.

[78] Nal Kalchbrenner, Edward Grefenstette, and Phil Blunsom. “A Convolutional Neural Network for Modelling Sentences.” In: *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 2014, pp. 655–665.

[79] Vishal Kaushal, Rishabh Iyer, Suraj Kothawade, Rohan Mahadev, Khoshrav Doctor, and Ganesh Ramakrishnan. “Learning from less data: A unified data subset selection and active learning framework for computer vision.” In: *2019 IEEE Winter Conference on Applications of Computer Vision (WACV)*. IEEE. 2019, pp. 1289–1299.

[80] Krishnateja Killamsetty, S Durga, Ganesh Ramakrishnan, Abir De, and Rishabh Iyer. “Grad-match: Gradient matching based data subset selection for efficient deep model training.” In: *International Conference on Machine Learning*. PMLR. 2021, pp. 5464–5474.

[81] Jin-Hwa Kim, Jaehyun Jun, and Byoung-Tak Zhang. “Bilinear attention networks.” In: *Advances in Neural Information Processing Systems*. 2018, pp. 1564–1574.

[82] Diederik P. Kingma and Jimmy Ba. “Adam: A Method for Stochastic Optimization.” In: *International Conference on Learning Representations*. Ed. by Yoshua Bengio and Yann LeCun. 2015.

[83] R. Koncel-Kedziorski, Hannaneh Hajishirzi, and Ali Farhadi. “Multi-resolution language grounding with weak supervision.” In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*. Oct. 2014, pp. 386–396. DOI: 10.3115/v1/D14-1043. URL: https://www.aclweb.org/anthology/D14-1043.

[84] Ioannis Konstas, Srinivasan Iyer, Mark Yatskar, Yejin Choi, and Luke Zettlemoyer. “Neural AMR: Sequence-to-Sequence Models for Parsing and Generation.” In: *Proceedings of the 55th Annual Meeting of the ACL*. Vancouver, Canada, 2017, pp. 146–157. DOI: 10.18653/v1/P17-1014. URL: https://www.aclweb.org/anthology/P17-1014.

[85] Ksenia Konyushkova, Raphael Sznitman, and Pascal Fua. "Introducing geometry in active learning for image segmentation." In: *Proceedings of the IEEE International Conference on Computer Vision*. 2015, pp. 2974–2982.

[86] Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A Shamma, et al. "Visual genome: Connecting language and vision using crowdsourced dense image annotations." In: *International journal of computer vision* 123.1 (2017).

[87] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. "Imagenet classification with deep convolutional neural networks." In: *Advances in neural information processing systems* 25 (2012).

[88] Karen Kukich. "Design of a knowledge-based report generator." In: *21st Annual Meeting of the Association for Computational Linguistics*. 1983, pp. 145–150.

[89] Dhruv Kumar, Lili Mou, Lukasz Golab, and Olga Vechtomova. "Iterative edit-based unsupervised sentence simplification." In: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. 2020, pp. 7918–7928.

[90] Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. "Neural Architectures for Named Entity Recognition." In: *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 2016, pp. 260–270.

[91] Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. "Albert: A lite bert for self-supervised learning of language representations." In: *arXiv preprint arXiv:1909.11942* (2019).

[92] Irene Langkilde and Kevin Knight. "Generation that exploits corpus-based statistical knowledge." In: *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics, Volume 1*. Aug. 1998, pp. 704–710. URL: https://www.aclweb.org/anthology/P98-1116.

[93] Agata Lapedriza, Hamed Pirsiavash, Zoya Bylinskii, and Antonio Torralba. "Are all training examples equally valuable?" In: *arXiv preprint arXiv:1311.6510* (2013).

[94] Alon Lavie and Abhaya Agarwal. "METEOR: An automatic metric for MT evaluation with high levels of correlation with human judgments." In: *Proceedings of the Second Workshop on Statistical Machine Translation*. June 2007, pp. 228–231. URL: https://www.aclweb.org/anthology/W07-0734.

[95] Rémi Lebret, David Grangier, and Michael Auli. "Neural text generation from structured data with application to the biography domain." In: *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. Nov. 2016, pp. 1203–1213. DOI: 10.18653/v1/D16-1128. URL: https://www.aclweb.org/anthology/D16-1128.

[96] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. "BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension." In: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. 2020, pp. 7871–7880.

[97] Jingjing Li, Zichao Li, Lili Mou, Xin Jiang, Michael R Lyu, and Irwin King. "Unsupervised text generation by learning from search." In: *Advances in Neural Information Processing Systems*. 2020. URL: https://papers.nips.cc/paper/2020/file/7a677bb4477ae2dd371add568dd19e23-Paper.pdf.

[98] Juncen Li, Robin Jia, He He, and Percy Liang. "Delete, retrieve, generate: A simple approach to sentiment and style transfer." In: *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*. 2018, pp. 1865–1874. URL: https://www.aclweb.org/anthology/N18-1169.

[99] Zichao Li, Xin Jiang, Lifeng Shang, and Hang Li. "Paraphrase Generation with Deep Reinforcement Learning." In: *Proceedings of the 2018 Conference on EMNLP*. Brussels, Belgium, 2018, pp. 3865–3878. DOI: 10.18653/v1/D18-1421. URL: https://www.aclweb.org/anthology/D18-1421.

[100] Zichao Li, Xin Jiang, Lifeng Shang, and Qun Liu. "Decomposable Neural Paraphrase Generation." In: *Proceedings of the 57th Annual Meeting of the ACL*. Florence, Italy, 2019, pp. 3403–3414. DOI: 10.18653/v1/P19-1332. URL: https://www.aclweb.org/anthology/P19-1332.

[101] Percy Liang, Michael Jordan, and Dan Klein. "Learning semantic correspondences with less supervision." In: *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*. Aug. 2009, pp. 91–99. URL: https://www.aclweb.org/anthology/P09-1011.

[102] Chin-Yew Lin. "ROUGE: A package for automatic evaluation of summaries." In: *Text Summarization Branches Out*. July 2004, pp. 74–81. URL: https://www.aclweb.org/anthology/W04-1013.

[103] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. "Focal loss for dense object detection." In: *Proceedings of the IEEE International Conference on Computer Vision*. 2017.

[104] Pengfei Liu, Xipeng Qiu, and Xuanjing Huang. "Recurrent neural network for text classification with multi-task learning." In: *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*. 2016, pp. 2873–2879.

[105] Tianyu Liu, Fuli Luo, Qiaolin Xia, Shuming Ma, Baobao Chang, and Zhifang Sui. "Hierarchical encoder with auxiliary supervision for neural table-to-text generation: Learning better representation for tables." In: *Proceedings of the AAAI Conference on Artificial Intelligence*. 2019, pp. 6786–6793. URL: https://ojs.aaai.org/index.php/AAAI/article/view/4653.

[106] Tianyu Liu, Kexiang Wang, Lei Sha, Baobao Chang, and Zhifang Sui. "Table-to-text generation by structure-aware seq2seq learning." In: *Thirty-Second AAAI Conference on Artificial Intelligence*. 2018.

[107] Xianggen Liu, Lili Mou, Fandong Meng, Hao Zhou, Jie Zhou, and Sen Song. "Unsupervised paraphrasing by simulated annealing." In: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. July 2020, pp. 302–312. DOI: 10.18653/v1/2020.acl-main.28. URL: https://www.aclweb.org/anthology/2020.acl-main.28.

[108] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. "Roberta: A robustly optimized bert pretraining approach." In: *arXiv preprint arXiv:1907.11692* (2019).

[109] Yuzong Liu, Rishabh Iyer, Katrin Kirchhoff, and Jeff Bilmes. "SVitchboard II and FiSVer I: High-quality limited-complexity corpora of conversational English speech." In: *Sixteenth Annual Conference of the International Speech Communication Association*. 2015.

[110] Ilya Loshchilov and Frank Hutter. "Decoupled weight decay regularization." In: *International Conference on Learning Representations*. 2018. URL: https://openreview.net/forum?id=Bkg6RiCqY7.

[111] Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. "ViLBERT: Pretraining Task-Agnostic Visiolinguistic Representations for Vision-and-Language Tasks." In: *Advances in Neural Information Processing Systems 32*. Ed. by H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett. Curran Associates, Inc., 2019, pp. 13–23. URL: http://papers.nips.cc/paper/8297-vilbert-pretraining-task-agnostic-

visiolinguistic-representations-for-vision-and-language-tasks.pdf.

[112] Jiasen Lu, Jianwei Yang, Dhruv Batra, and Devi Parikh. "Hierarchical question-image co-attention for visual question answering." In: *Advances in neural information processing systems* 29 (2016).

[113] Mario Lucic, Matthew Faulkner, Andreas Krause, and Dan Feldman. "Training gaussian mixture models at scale via coresets." In: *The Journal of Machine Learning Research* 18.1 (2017), pp. 5885–5909.

[114] Shuming Ma, Pengcheng Yang, Tianyu Liu, Peng Li, Jie Zhou, and Xu Sun. "Key Fact as Pivot: A Two-Stage Model for Low Resource Table-to-Text Generation." In: *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Florence, Italy: Association for Computational Linguistics, July 2019, pp. 2047–2057. DOI: 10.18653/v1/P19-1197. URL: https://aclanthology.org/P19-1197.

[115] Nikolaos Malandrakis, Minmin Shen, Anuj Goyal, Shuyang Gao, Abhishek Sethi, and Angeliki Metallinou. "Controlled Text Generation for Data Augmentation in Intelligent Artificial Agents." In: *Proceedings of the 3rd Workshop on Neural Generation and Translation*. Hong Kong, 2019, pp. 90–98. DOI: 10.18653/v1/D19-5609. URL: https://www.aclweb.org/anthology/D19-5609.

[116] Mateusz Malinowski and Mario Fritz. "A multi-world approach to question answering about real-world scenes based on uncertain input." In: *Advances in neural information processing systems*. 2014, pp. 1682–1690.

[117] Mateusz Malinowski and Mario Fritz. "Towards a visual turing challenge." In: *arXiv preprint arXiv:1410.8027* (2014).

[118] Tom McCoy, Ellie Pavlick, and Tal Linzen. "Right for the Wrong Reasons: Diagnosing Syntactic Heuristics in Natural Language Inference." In: *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Florence, Italy: Association for Computational Linguistics, July 2019, pp. 3428–3448. DOI: 10.18653/v1/P19-1334. URL: https://www.aclweb.org/anthology/P19-1334.

[119] Warren S McCulloch and Walter Pitts. "A logical calculus of the ideas immanent in nervous activity." In: *The bulletin of mathematical biophysics* 5.4 (1943), pp. 115–133.

[120] David D McDonald. "Issues in the choice of a source for natural language generation." In: *Computational Linguistics* 19.1 (1993), pp. 191–197.

[121] Kathleen McKeown. *Text Generation*. Cambridge University Press, 1992. URL: https://www.cambridge.org/ca/academic/subjects/languages-linguistics/computational-linguistics/text-generation?format=PB&isbn=9780521438025.

[122] Hongyuan Mei, Mohit Bansal, and Matthew R Walter. "What to talk about and how? Selective Generation using LSTMs with Coarse-to-Fine Alignment." In: *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 2016, pp. 720–730.

[123] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. "Distributed representations of words and phrases and their compositionality." In: *Advances in neural information processing systems* 26 (2013).

[124] Tomáš Mikolov, Édouard Grave, Piotr Bojanowski, Christian Puhrsch, and Armand Joulin. "Advances in Pre-Training Distributed Word Representations." In: *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*. 2018.

[125] Mike Mintz, Steven Bills, Rion Snow, and Daniel Jurafsky. "Distant supervision for relation extraction without labeled data." In: *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*. Suntec, Singapore: Association for Computational Linguistics, Aug. 2009, pp. 1003–1011. URL: https://aclanthology.org/P09-1113.

[126] Riccardo Miotto, Fei Wang, Shuang Wang, Xiaoqian Jiang, and Joel T Dudley. "Deep learning for healthcare: review, opportunities and challenges." In: *Briefings in bioinformatics* 19.6 (2018), pp. 1236–1246.

[127] Baharan Mirzasoleiman, Jeff Bilmes, and Jure Leskovec. "Coresets for data-efficient training of machine learning models." In: *International Conference on Machine Learning*. PMLR. 2020, pp. 6950–6960.

[128] Manish Munikar, Sushil Shakya, and Aakash Shrestha. "Fine-grained sentiment classification using bert." In: *2019 Artificial Intelligence for Transforming Business and Society (AITB)*. Vol. 1. IEEE. 2019, pp. 1–5.

[129] Deepak Muralidharan et al. "Leveraging User Engagement Signals For Entity Labeling in a Virtual Assistant." In: *arXiv* 1909.09143 (2019).

[130] Kuniaki Noda, Yuki Yamaguchi, Kazuhiro Nakadai, Hiroshi G Okuno, and Tetsuya Ogata. "Audio-visual speech recognition using deep learning." In: *Applied Intelligence* 42.4 (2015), pp. 722–737.

[131] Jekaterina Novikova, Ondřej Dušek, and Verena Rieser. "The E2E dataset: New challenges for end-to-end generation." In: *Proceedings of the 18th Annual SIGdial Meeting on Discourse and Dialogue*. Aug. 2017, pp. 201–206. DOI: 10.18653/v1/W17-5525. URL: https://www.aclweb.org/anthology/W17-5525.

[132] Franz Josef Och and Hermann Ney. "A systematic comparison of various statistical alignment models." In: *Computational linguistics* 29.1 (2003), pp. 19–51.

[133] 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*. 2002, pp. 311–318. URL: https://www.aclweb.org/anthology/P02-1040.

[134] Ankur P Parikh, Xuezhi Wang, Sebastian Gehrmann, Manaal Faruqui, Bhuwan Dhingra, Diyi Yang, and Dipanjan Das. "ToTTo: A Controlled Table-To-Text Generation Dataset." In: *Proceedings of EMNLP*. 2020.

[135] Ajeet Ram Pathak, Manjusha Pandey, and Siddharth Rautaray. "Application of deep learning for object detection." In: *Procedia computer science* 132 (2018), pp. 1706–1717.

[136] Jeffrey Pennington, Richard Socher, and Christopher Manning. "Glove: Global Vectors for Word Representation." In: *Proceedings of the 2014 Conference on EMNLP*. Doha, 2014, pp. 1532–1543.

[137] Aaditya Prakash, Sadid A. Hasan, Kathy Lee, Vivek Datla, Ashequl Qadir, Joey Liu, and Oladimeji Farri. "Neural Paraphrase Generation with Stacked Residual LSTM Networks." In: *Proceedings of the 26th International Conference on Computational Linguistics*. Osaka, Japan, 2016, pp. 2923–2934. URL: https://www.aclweb.org/anthology/C16-1275.

[138] Xipeng Qiu, Tianxiang Sun, Yige Xu, Yunfan Shao, Ning Dai, and Xuanjing Huang. "Pre-trained models for natural language processing: A survey." In: *Science China Technological Sciences* 63.10 (2020), pp. 1872–1897.

[139] Zimeng Qiu, Eunah Cho, Xiaochun Ma, and William Campbell. "Graph-Based Semi-Supervised Learning for Natural Language Understanding." In: *Proceedings of the Thirteenth Workshop on Graph-Based Methods for NLP*. Hong Kong, 2019, pp. 151–158. DOI: 10.18653/v1/D19-5318. URL: https://www.aclweb.org/anthology/D19-5318.

[140] Chen Qu, Liu Yang, Minghui Qiu, W Bruce Croft, Yongfeng Zhang, and Mohit Iyyer. "BERT with history answer embedding for conversational question answering." In: *Proceedings of the 42nd international ACM SIGIR conference on research and development in information retrieval*. 2019, pp. 1133–1136.

[141] Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. "Improving language understanding by generative pre-training." In: *URL https://s3-us-west-2. amazonaws. com/openai-assets/researchcovers/languageunsupervised/language understanding paper. pdf* (2018).

[142] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. "Language models are unsupervised multitask learners." In: ().

[143] 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." In: *Journal of Machine Learning Research* 21 (2020), pp. 1–67.

[144] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. "Squad: 100,000+ questions for machine comprehension of text." In: *arXiv preprint arXiv:1606.05250* (2016).

[145] Nils Reimers and Iryna Gurevych. "Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks." In: *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Nov. 2019. URL: http://arxiv.org/abs/1908.10084.

[146] Ehud Reiter and Robert Dale. "Building applied natural language generation systems." In: *Natural Language Engineering* 3.1 (1997), pp. 57–87.

[147] Mengye Ren, Ryan Kiros, and Richard Zemel. "Exploring models and data for image question answering." In: *Advances in neural information processing systems*. 2015, pp. 2953–2961.

[148] Sebastian Riedel, Limin Yao, and Andrew McCallum. "Modeling relations and their mentions without labeled text." In: *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer. 2010, pp. 148–163.

[149] Verena Rieser and Oliver Lemon. "Natural language generation as planning under uncertainty for spoken dialogue systems." In: *Proceedings of the 12th Conference of the European Chapter of the ACL*. Mar. 2009, pp. 683–691. URL: https://www.aclweb.org/anthology/E09-1078.

[150] Alexander M Rush, Sumit Chopra, and Jason Weston. “A Neural Attention Model for Abstractive Sentence Summarization.” In: *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. 2015, pp. 379–389.

[151] Greg Schohn and David Cohn. “Less is more: Active learning with support vector machines.” In: *ICML*. Vol. 2. 4. Citeseer. 2000, p. 6.

[152] Florian Schroff, Dmitry Kalenichenko, and James Philbin. “Facenet: A unified embedding for face recognition and clustering.” In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015, pp. 815–823.

[153] Raphael Schumann, Lili Mou, Yao Lu, Olga Vechtomova, and Katja Markert. “Discrete optimization for unsupervised sentence summarization with word-level extraction.” In: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. July 2020, pp. 5032–5042. URL: https://www.aclweb.org/anthology/2020.acl-main.452.

[154] 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 ACL*. Vancouver, Canada, 2017, pp. 1073–1083. DOI: 10.18653/v1/P17-1099. URL: https://www.aclweb.org/anthology/P17-1099.

[155] Lei Sha, Lili Mou, Tianyu Liu, Pascal Poupart, Sujian Li, Baobao Chang, and Zhifang Sui. “Order-planning neural text generation from structured data.” In: *Thirty-Second AAAI Conference on Artificial Intelligence*. 2018.

[156] Lei Sha, Lili Mou, Tianyu Liu, Pascal Poupart, Sujian Li, Baobao Chang, and Zhifang Sui. “Order-planning neural text generation from structured data.” In: *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence*. 2018, pp. 5414–5421.

[157] Meet Shah, Xinlei Chen, Marcus Rohrbach, and Devi Parikh. “Cycle-consistency for robust visual question answering.” In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2019, pp. 6649–6658.

[158] Ali Sharif Razavian, Hossein Azizpour, Josephine Sullivan, and Stefan Carlsson. “CNN features off-the-shelf: an astounding baseline for recognition.” In: *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*. 2014, pp. 806–813.

[159] Sheng Shen, Daniel Fried, Jacob Andreas, and Dan Klein. “Pragmatically informative text generation.” In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. June 2019, pp. 4060–4067. DOI:

10.18653/v1/N19-1410. URL: https://www.aclweb.org/anthology/N19-1410.

[160] Connor Shorten and Taghi M Khoshgoftaar. "A survey on image data augmentation for deep learning." In: *Journal of big data* 6.1 (2019), pp. 1–48.

[161] Advaith Siddharthan. "A survey of research on text simplification." In: *ITL-International Journal of Applied Linguistics* 165.2 (2014), pp. 259–298.

[162] Karen Simonyan and Andrew Zisserman. "Very deep convolutional networks for large-scale image recognition." In: *arXiv preprint arXiv:1409.1556* (2014).

[163] Shashi Pal Singh, Ajai Kumar, Hemant Darbari, Lenali Singh, Anshika Rastogi, and Shikha Jain. "Machine translation using deep learning: An overview." In: *2017 international conference on computer, communications and electronics (comptelix)*. IEEE. 2017, pp. 162–167.

[164] Richard Socher, Jeffrey Pennington, Eric H Huang, Andrew Y Ng, and Christopher D Manning. "Semi-supervised recursive autoencoders for predicting sentiment distributions." In: *Proceedings of the 2011 conference on empirical methods in natural language processing*. 2011, pp. 151–161.

[165] Harold Somers. "An introduction to machine translation." In: (1992).

[166] Amanda Stent, Rashmi Prasad, and Marilyn Walker. "Trainable sentence planning for complex information presentations in spoken dialog systems." In: *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics*. 2004, pp. 79–86. URL: https://www.aclweb.org/anthology/P04-1011.

[167] Alane Suhr, Mike Lewis, James Yeh, and Yoav Artzi. "A Corpus of Natural Language for Visual Reasoning." In: *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Short Papers)*. 2017.

[168] Mihai Surdeanu, Julie Tibshirani, Ramesh Nallapati, and Christopher D. Manning. "Multi-instance Multi-label Learning for Relation Extraction." In: *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*. Jeju Island, Korea: Association for Computational Linguistics, July 2012, pp. 455–465. URL: https://aclanthology.org/D12-1042.

[169] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. "Sequence to sequence learning with neural networks." In: *Advances in neural information processing systems* 27 (2014).

[170] Hao Tan and Mohit Bansal. "LXMERT: Learning Cross-Modality Encoder Representations from Transformers." 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)*. Hong Kong, China: Association for Computational Linguistics, Nov. 2019, pp. 5100–5111. DOI: 10.18653/v1/D19-1514. URL: https://www.aclweb.org/anthology/D19-1514.

[171] Damien Teney, Peter Anderson, Xiaodong He, and Anton Van Den Hengel. "Tips and tricks for visual question answering: Learnings from the 2017 challenge." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018.

[172] Kento Terao, Toru Tamaki, Bisser Raytchev, Kazufumi Kaneda, and Shun'ichi Satoh. "Which visual questions are difficult to answer? Analysis with Entropy of Answer Distributions." In: *arXiv preprint arXiv:2004.05595* (2020).

[173] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. "Attention is all you need." In: *Advances in neural information processing systems* 30 (2017).

[174] Ramakrishna Vedantam, C Lawrence Zitnick, and Devi Parikh. "CIDEr: Consensus-based image description evaluation." In: *Proceedings of the 2015 IEEE Conference on Computer Vision and Pattern Recognition*. 2015, pp. 4566–4575. URL: https://openaccess.thecvf.com/content_cvpr_2015/papers/Vedantam_CIDEr_Consensus-Based_Image_2015_CVPR_paper.pdf.

[175] Oriol Vinyals and Quoc Le. "A neural conversational model." In: *arXiv preprint arXiv:1506.05869* (2015).

[176] Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. "Show and tell: A neural image caption generator." In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015, pp. 3156–3164.

[177] Athanasios Voulodimos, Nikolaos Doulamis, Anastasios Doulamis, and Eftychios Protopapadakis. "Deep learning for computer vision: A brief review." In: *Computational intelligence and neuroscience* 2018 (2018).

[178] Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. "GLUE: A Multi-Task Benchmark and Analysis Platform for Natural Language Understanding." In: *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*. Brussels, Belgium: Association for Computational Linguistics, Nov. 2018, pp. 353–355. DOI: 10.18653/v1/W18-5446. URL: https://www.aclweb.org/anthology/W18-5446.

[179] Hao Wang, Bing Liu, Chaozhuo Li, Yan Yang, and Tianrui Li. "Learning with Noisy Labels for Sentence-level Sentiment Classification." 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)*. Hong Kong, China: Association for Computational Linguistics, Nov. 2019, pp. 6286–6292. DOI: 10.18653/v1/D19-1655. URL: https://aclanthology.org/D19-1655.

[180] Jianyu Wang, Zhishuai Zhang, Cihang Xie, Yuyin Zhou, Vittal Premachandran, Jun Zhu, Lingxi Xie, and Alan Yuille. "Visual concepts and compositional voting." In: *Annals of Mathematical Sciences and Applications* 3.1 (2018), pp. 151–188. ISSN: 2380-2898. DOI: 10.4310/amsa.2018.v3.n1.a5. URL: http://dx.doi.org/10.4310/amsa.2018.v3.n1.a5.

[181] Xin Wang, Yuanchao Liu, Cheng-Jie Sun, Baoxun Wang, and Xiaolong Wang. "Predicting polarities of tweets by composing word embeddings with long short-term memory." In: *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. 2015, pp. 1343–1353.

[182] Yequan Wang, Minlie Huang, Xiaoyan Zhu, and Li Zhao. "Attention-based LSTM for aspect-level sentiment classification." In: *Proceedings of the 2016 conference on empirical methods in natural language processing*. 2016, pp. 606–615.

[183] Yunli Wang, Yu Wu, Lili Mou, Zhoujun Li, and Wenhan Chao. "Harnessing pre-trained neural networks with rules for formality style transfer." In: *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*. 2019, pp. 3573–3578.

[184] Jason Wei and Kai Zou. "EDA: Easy Data Augmentation Techniques for Boosting Performance on Text Classification Tasks." 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)*. Hong Kong, China: Association for Computational Linguistics, Nov. 2019, pp. 6382–6388. DOI: 10.18653/v1/D19-1670. URL: https://aclanthology.org/D19-1670.

[185] Kai Wei, Yuzong Liu, Katrin Kirchhoff, and Jeff Bilmes. "Unsupervised submodular subset selection for speech data." In: *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE. 2014, pp. 4107–4111.

[186] Paul J Werbos. “Backpropagation through time: what it does and how to do it.” In: *Proceedings of the IEEE* 78.10 (1990), pp. 1550–1560.

[187] Sam Wiseman, Stuart Shieber, and Alexander Rush. “Learning neural templates for text generation.” In: *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. 2018, pp. 3174–3187. DOI: 10.18653/v1/D18-1356. URL: https://www.aclweb.org/anthology/D18-1356.

[188] Caiming Xiong, Stephen Merity, and Richard Socher. “Dynamic memory networks for visual and textual question answering.” In: *International conference on machine learning*. PMLR. 2016, pp. 2397–2406.

[189] Huijuan Xu and Kate Saenko. “Ask, attend and answer: Exploring question-guided spatial attention for visual question answering.” In: *European Conference on Computer Vision*. Springer. 2016, pp. 451–466.

[190] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. “Show, attend and tell: Neural image caption generation with visual attention.” In: *International conference on machine learning*. PMLR. 2015, pp. 2048–2057.

[191] Chun-Ju Yang, Kristen Grauman, and Danna Gurari. “Visual question answer diversity.” In: *Proceedings of the AAAI Conference on Human Computation and Crowdsourcing*. Vol. 6. 1. 2018.

[192] Zichao Yang, Xiaodong He, Jianfeng Gao, Li Deng, and Alex Smola. “Stacked attention networks for image question answering.” In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 21–29.

[193] Tom Young, Devamanyu Hazarika, Soujanya Poria, and Erik Cambria. “Recent trends in deep learning based natural language processing.” In: *ieee Computational intelligenCe magazine* 13.3 (2018), pp. 55–75.

[194] Licheng Yu, Eunbyung Park, Alexander C Berg, and Tamara L Berg. “Visual madlibs: Fill in the blank description generation and question answering.” In: *Proceedings of the ieee international conference on computer vision*. 2015, pp. 2461–2469.

[195] Zhou Yu, Jun Yu, Jianping Fan, and Dacheng Tao. “Multimodal Factorized Bilinear Pooling with Co-attention Learning for Visual Question Answering.” In: *2017 IEEE International Conference on Computer Vision (ICCV)* (2017). DOI: 10.1109/iccv.2017.202. URL: http://dx.doi.org/10.1109/ICCV.2017.202.

[196] Peng Zhang, Yash Goyal, Douglas Summers-Stay, Dhruv Batra, and Devi Parikh. “Yin and yang: Balancing and answering binary visual questions.” In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2016, pp. 5014–5022.

[197] Peng Zhang, Yash Goyal, Douglas Summers-Stay, Dhruv Batra, and Devi Parikh. “Yin and yang: Balancing and answering binary visual questions.” In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2016.

[198] Ye Zhang and Byron Wallace. “A sensitivity analysis of (and practitioners’ guide to) convolutional neural networks for sentence classification.” In: *arXiv preprint arXiv:1510.03820* (2015).

[199] Luowei Zhou, Hamid Palangi, Lei Zhang, Houdong Hu, Jason Corso, and Jianfeng Gao. “Unified vision-language pretraining for image captioning and vqa.” In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 34. 07. 2020, pp. 13041–13049.

[200] Xiaojin Zhu and Andrew B Goldberg. “Introduction to semi-supervised learning.” In: *Synthesis Lectures on Artificial Intelligence and Machine Learning* 3.1 (2009), pp. 1–130. URL: https://www.morganclaypool.com/doi/abs/10.2200/S00196ED1V01Y200906AIM006.

[201] Yuke Zhu, Oliver Groth, Michael Bernstein, and Li Fei-Fei. “Visual7w: Grounded question answering in images.” In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2016.

[202] Yukun Zhu, Ryan Kiros, Rich Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. “Aligning books and movies: Towards story-like visual explanations by watching movies and reading books.” In: *Proceedings of the IEEE international conference on computer vision*. 2015, pp. 19–27.

SHAILZA JOLLY

EDUCATION

TU Kaiserslautern Germany March 2019 - Novemeber 2022
P.hD. in Computer Science
Developing machine learning methods for building natural language generation and understanding systems in data constrained settings. Also interested in vision and language, interpretability, and conversational AI.
Overall grade: **"Sehr Gut - Magna Cum Laude" 1.0** (1.0 is highest on 1.0 - 5.0 scale)

TU Kaiserslautern, Germany October 2016 - November 2018
MSc. in Computer Science | Minor in Economics
Thesis: An Evaluation Look at the Evaluation of VQA: The wisdom of MaSSeS **(Grade: 1.0)**
Overall grade: **"Sehr Gut" 1.5**

Guru Nanak Dev Engineering College, India August 2012 - July 2016
B.Tech in Computer Science & Engineering
Overall grade: **First Division with Distinction**

WORK EXPERIENCE

Alexa AI, Amazon July 2022 - Present
Research Scientist *Berlin, Germany*

· Working on language understanding in Amazon Alexa.

Deutsche Forschungszentrum für Künstliche Intelligenz Gmbh March 2019 - June 2022
Research Assistant *Berlin, Germany*

· Working on BMBF funded project XAINES that aims explainability of Machine Learning and Artificial Intelligence systems (started October 2020).
· Worked on BMBF funded project Deep Fusion for Neural Networks (DeFuseNN) in the area of understanding vision and language systems (ended September 2020).

NVIDIA Research (Learning and Perception Research) May - August 2021
Machine Learning Scientist Intern *Santa Clara, United States*

· Worked with Dr. Thomas Breuel on information extraction from financial documents.

University of Copenhagen January 2021 - March 2021
Visiting Researcher *Copenhagen, Denmark*

· Worked with Prof. Isabelle Augenstein on building explainable fact-checking systems using Natural Language Generation.

University of Alberta September 2020 - December 2020
Visiting Researcher *Alberta, Canada*

· Worked with Prof. Lili Mou on Natural Language Generation using scoring based methods.

Amazon Alexa August 2019 - December 2019
Applied Scientist Intern *Aachen, Germany*

· Worked with Dr. Caglar Tirkaz and Dr. Tobias Falke on Natural Language Generation in context of paraphrasing to improve NLU systems with data augmentation.

SAP AI Research May 2018 - February 2019
Research Intern/ Master Thesis *Berlin, Germany*

· Worked with Dr. Moin Nabi and Dr. Tassilo Klein on designing an evaluation metric called MaSSeS for Visual Question Answering models.

Human Interface Laboratory, Kyushu University November 2017 - February 2018
Visiting Researcher *Fukuoka, Japan*

· Worked with Prof. Seiichi Uchida on project "Explainable AI" to analyze the behavior of deep CNN architectures for image recognition.

Deutsche Forschungszentrum für Künstliche Intelligenz Gmbh February 2017 - 2018
Student Research Assistant *Kaiserslautern, Germany*

· Worked with Dr. Damian Borth on social network analysis for continuous monitoring of corporate twitter accounts.

REFEREED PUBLICATIONS

- **Shailza Jolly**, Pepa Atanasova, Isabelle Augenstein. Generating Fluent Fact Checking Explanations with Unsupervised Post-Editing. In Information, 13(10), 500, MDPI Journal.
- **Shailza Jolly**, Zi Xuan Zhang, Andreas Dengel, Lili Mou (2022). Search and Learn: Improving Semantic Coverage for Data-to-Text Generation. In Proceedings of the Thirty-Sixth AAAI Conference on Artificial Intelligence (AAAI 2022), February 2022. (*Acceptance Rate: 15%*).
- The GEM Benchmark: Natural Language Generation, its Evaluation and Metrics. In *Proceedings* of the 1st Workshop on Natural Language Generation, its Evaluation and Metrics at *ACL* 2021.
- **Shailza Jolly**, Sandro Pezzelle, Moin Nabi (2021). EaSe: A Diagnostic Tool for VQA based on Answer Diversity. In *Proceedings of NAACL-HLT* 2021, June 2021 (*Acceptance Rate: 26%*).
- **Shailza Jolly**, Tobias Falke, Caglar Tirkaz, Daniil Sorokin (2020). Data-Efficient Paraphrase Generation to Bootstrap Intent Classification and Slot Labeling for New Features in Task-Oriented Dialog Systems. In *Proceedings of COLING* 2020, December 2020 (*Industry Track*) (*Acceptance Rate: 22.9%*).
- **Shailza Jolly**, Shubham Kapoor (2020). Can Pre-training help VQA with Lexical Variations? In *Proceedings of EMNLP Findings* 2020, November 2020 (*Acceptance Rate: 15.4%).*
- Stanislav Frolov, **Shailza Jolly**, Joern Hees, Andreas Dengel (2020). Leveraging Visual Question Answering to Improve Text-to-Image Synthesis. In *Proceedings* of Second Workshop Beyond Vision and LANguage: inTEgrating Real-world kNowledge (LANTERN) at *COLING* 2020.
- **Shailza Jolly***, Sebastian Palacio*, Joachim Folz, Federico Raue, Joern Hees, Andreas Dengel (2020). P $\approx$ NP, at least in Visual Question Answering. In *Proceedings of ICPR* 2020, January 2021 (*Acceptance Rate: 36%*).
- **Shailza Jolly**, Brian Kenji Iwana, Ryohei Kuroki, Seiichi Uchida (2018). How do Convolutional Neural Networks Learn Design?. In *Proceedings of International Conference on Pattern Recognition (ICPR)* 2018, August 2018 (*Best Student Paper*) .

PREPRINTS

- **Shailza Jolly**, Andreas Dengel, Lili Mou (2021). Search and Learn: Improving Semantic Coverage for Data-to-Text Generation. (*https://tinyurl.com/uzwnds*)
- **Shailza Jolly**, Sandro Pezzelle, Tassilo Klein, Andreas Dengel, Moin Nabi (2018). The Wisdom of MaSSeS: Majority, Subjectivity, and Semantic Similarity in the Evaluation of VQA. (*https://arxiv.org/abs/1809.04344*)

AWARDS AND HONORS

- Received **AAAI-22 Scholarship** by Hitachi to attend AAAI 2022 conference (2022).
- Awarded **AI Newcomer 2021 Award** for computer science discipline by German Informatics Society and the Federal Ministry of Education and Research (BMBF), Germany (2021).
- Received **EU-Cost STSM Grant** to work on Multi3generation project at CopeNLU (2020).
- Awarded a **travel grant** to attend Amazon AI Research Colloquium at Cambridge UK (2019).
- **3rd position** at the TextVQA challenge at International Conference on Document Analysis and Recognition (ICDAR) (2019).
- Awarded **Ph.D. Research Fellowship** by TU Kaiserslautern, Germany.
- Awarded a **travel grant** to attend Pre-Doctoral School at Max Planck Institute for Intelligent Systems, Tuebingen, Germany (2019).
- **Best Student Paper Award** at ICPR 2018 along with cash prize of USD 500 (2018).
- Awarded a **travel grant** to work at Kyushu University, Japan (2017).
- Awarded a **travel grant** to attend Deep Learning Summer School, Bilbao (2017).
- Awarded **academic scholarship** worth INR 10000 in All India Secondary School Exam (2010).
- **Among top 0.1% students** for National level Class 10 Maths Exam (A1 Grade) (2010).

INVITED TALKS / PRESENTATIONS

May 2022	Poster Presentation at DFKI-Review meeting.
February 2022	Oral & Poster Presentation at AAAI 2022.
June 2021	Oral & Poster Presentation at NAACL 2021.
March 2021	Invited talk at CopeNLU, Denmark.
January 2021	Invited talk at the Intel Labs, USA.
January 2021	Poster presentation at ICPR 2020.
October 2020	Invited talk at the Language Technology Lab, Saarland Germany.
October 2020	Invited talk at the University of Alberta, Canada.
October 2020	Oral Presentation at COLING 2020.
October 2020	Oral Presentation at LANTERN workshop, COLING 2020.
August 2019	Poster Presentation at Amazon AI Research Colloquium, Cambridge UK.
September 2018	Poster Presentation at Workshop on Shortcomings in Vision and Language (SiVL), European Conference on Computer Vision (ECCV), 2018
August 2018	Oral Presentation at ICPR 2018

SERVICE ROLES AND ACADEMIC ACTIVITIES

- Served on the program committee for AAAI-23.
- Selected as volunteer at EMNLP 2021, ACL 2021, NAACL 2021, EMNLP 2020, ICML 2020.
- Served in the program committee of First & Second Workshop on Advances in Language and Vision Research (ALVR) at ACL 2020 and NAACL 2021.
- Served in the program committee of Natural Language Generation, its Evaluation and Metrics (GEM) workshop at ACL 2021.
- Sub-reviewer in IAPR International Workshop on Document Analysis Systems (DAS), 2020.

TEACHING

- TA in lecture series on "Applications of Machine Learning and Data Science" for WS 2021/22.
- Supervised **3** seminars (4 ECTS credits) and **1** master project (8 ECTS credits).

MEDIA COVERAGE

- Interview featured in the Rheinfalz newspaper.
- Radio interview for Antenne Kaiserslautern, Germany.