Statistical Natural Language Generation from Tabular Non-textual Data

A thesis
submitted in partial fulfillment of the requirement for the
Degree of
Master of Computer Science and Engineering
of
JADAVPUR UNIVERSITY
by

Joy Mahapatra
Registration Number : 129006 of 2014-15
Examination Roll Number : M4CSE1618

Under the Guidance of

Dr. Sudip Kumar Naskar and **Dr. Sivaji Bandyopadhyay**
Department of Computer Science and Engineering
Jadavpur University, Kolkata-700032
India

2016

FACULTY OF ENGINEERING AND TECHNOLOGY
JADAVPUR UNIVERSITY

Certificate of Recommendation

This is to certify that the dissertation entitled **"Statistical Natural Language Generation from Tabular Non-textual Data"** has been carried out by **Joy Mahapatra** (Registration No.: 129006 of 2014-15 , Examination Roll No.: M4CSE1618) under my guidance and supervision and be accepted in partial fulfillment of the requirement for the Degree of Master of Computer Science and Engineering. The research results presented in the thesis have not been included in any other thesis submitted for the award of any degree in any other University or Institute.

Dr. Sudip Kumar Naskar (Thesis Supervisor)
Department of Computer Science and Engineering
Jadavpur University

Dr. Sivaji Bandyopadhyay (Thesis Supervisor)
Department of Computer Science and Engineering
Jadavpur University

Dr. Debesh Das
Head, Department of Computer Science and Engineering
Jadavpur University

Dr. Sivaji Bandyopadhyay
Dean, Faculty of Engineering and Technology
Jadavpur University

FACULTY OF ENGINEERING AND TECHNOLOGY
JADAVPUR UNIVERSITY

Certificate of Approval[1]

This is to certify that the thesis entitled **"Statistical Natural Language Generation from Tabular Non-textual Data"** is a bona-fide record of work carried out by **Joy Mahapatra** in partial fulfillment of the requirements for the award of the degree of Master of Computer Science and Engineering in the Department of Computer Science and Engineering, Jadavpur University during the period June 2015 to May 2016. It is understood that by this approval the undersigned do not necessarily endorse or approve any statement made, opinion expressed or conclusion drawn therein but approve the thesis only for the purpose for which it has been submitted.

_____

Signature of Examiner 1

Date:

_____

Signature of Examiner 2

Date:

_____

Signature of Examiner 3

Date:

[1] Only in case the thesis is approved

Declaration of Authorship

I, **Joy Mahapatra**, declare that this thesis titled, **'Statistical Natural Language Generation from Tabular Non-textual Data'** and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed: _____

Date: _____

Abstract

The principal objective of Natural Language Generation (NLG) process is to produce human readable and understandable natural language text from different forms of non-textual data. There are many tremendous application areas of NLG and it evens growing day by day. From automatic customer-support systems to real-time weather report generation software, NLG technique plays a huge role in this modern digital world.

Presently many NLG systems are available in world. Although those systems can generate fluent and good natural text, but most of them either takes a long time or too much human work to do that text generation task.

In this thesis work, we present an automatic statistical natural language generation system which can convert tuple formed non-textual data into corresponding natural textual data without taking any human efforts. Proposed system doesnt required long time to produced textual document. The proposed NLG system needs an initial parallel (data/-text) corpus for training purpose. There are two major steps to build this system. At firstly a special kind of graph (Attribute Graph) needs to be created for storing corpuss data properly. Secondly some prediction through that graph has been made to generate natural language textual data. In our text generation system we have maintained both informativeness and linguistic quality of generated output text data. For holding informativeness property we make a prediction to incorporate more input non-textual data into output generated text. We also concern with linguistic quality of resulted text data with imposing language model on it.

We have evaluated our system with both automatic and human evaluation techniques. From both of those evaluation results we have found that our system occupy on a good rank compared to the other NLG systems. The results are even quite encouraging when we look the fact that, this system doesnt require any human effort compared to the others natural language systems...

Acknowledgements

First and foremost, I would like to thank my advisers Dr. Sudip Kumar Naskar and Dr. Sivaji Bandyopadhyay for their guidance and support during my research. They provided me with the opportunities to pursue my research interests and helped in developing a thinking processes required for the research.

I would also like to thank all my friends belonging to the Natural Language Processing Lab of Jadavpur University which include Manish Babu, Tapas Nayak, Alapan kulia, Somnath Banerjee, Pintu Lohar, Tanik Sk., Sandip Sarkar and Shouvik Roy from whom I learnt a lot about NLP.

My acknowledgment would not be complete without the mention of my classmates of PG CSE JU 2014 batch at Jadavpur University.

Contents

Declaration of Authorship iii

Abstract iv

Acknowledgements v

List of Figures viii

List of Tables ix

1 Introduction 1
- 1.1 Natural Language Generation (NLG) . . . 1
- 1.2 Application Areas of NLG . . . 1
- 1.3 Where NLG Systems Are Not Usable . . . 3
- 1.4 Inefficient Template Based Techniques . . . 3
- 1.5 Approaches for Construction of NLG Systems . . . 4
- 1.6 Subtasks of an NLG System . . . 4
  - 1.6.1 Content Determination . . . 5
  - 1.6.2 Discourse Planning . . . 6
  - 1.6.3 Sentence Aggregation . . . 6
- 1.7 Lexicalization . . . 7
- 1.8 Referring Expression Generation . . . 7
- 1.9 Sentence Realization . . . 8
- 1.10 Architecture of NLG Systems . . . 9
- 1.11 Problem with this NLG Architecture . . . 11
- 1.12 Our Proposed NLG system . . . 11
- 1.13 Contribution . . . 12
- 1.14 Structure Of This Thesis . . . 13

2 Related Works 14
- 2.1 Rule Based Techniques . . . 14
  - 2.1.1 SumTime System . . . 15
  - 2.1.2 FoG System . . . 15

2.1.3 PLANDOC System . . . 15
2.2 Handcrafted Grammar Based System . . . 15
2.2.1 Probabilistic Context Free Grammar Based NLG System . . . 16
2.3 Automatic Statistical Surface Realization System . . . 16
2.3.1 HPSG-based sentence realizer . . . 16
2.4 Semi-automatically Extracted Rule Based Generator . . . 17
2.4.1 Probabilistic Synchronous Context-Free Grammar Based Generator . . . 17
2.5 Fully Automatic Language Generation Systems . . . 17
2.5.1 Mountain System . . . 18
2.5.2 Generation Through Phrase-based Statistical Machine Translation 18
2.5.3 Case Based Natural Language Generation . . . 18
2.5.4 Neural Network Based Automatic Language Generator . . . 19

3 Problem Overview 20
3.1 Task Definition . . . 20
3.2 Requirement Analysis . . . 21
3.3 Design Strategy . . . 21

4 Development Phase (Text Generation) 23
4.1 Attribute Graph . . . 23
4.1.1 Intuition of Attribute Graph . . . 23
4.1.2 Building of Attribute Graph . . . 24
4.2 Text Generation Using Attribute Graph . . . 25
4.2.1 Informativeness Management Module . . . 26
4.2.2 Linguistic Quality Management Module . . . 30
4.2.3 Decoding . . . 30

5 Dataset and Evaluation 32
5.1 Dataset . . . 32
5.2 Evaluation . . . 33
5.2.1 Automatic Evaluation . . . 34
5.2.2 Human Evaluation . . . 35
5.3 Result Analysis . . . 36

6 Conclusions and Future Work 37
6.1 Conclusions . . . 37
6.2 Future Work . . . 38

Bibliography 39

List of Figures

1.1 Tasks of an NLG system 2
1.2 Application of NLG in Dialog System 2
1.3 An architecture for NLG system 10

3.1 Basic input-output structure of our system 21
3.2 Designing strategy behind our proposed model 22

4.1 Word lattice formation and generation of text 24
4.2 Attribute Graph 27

5.1 Transformation of Prodigy-METEO corpus non-textual data representation for our proposed system's input 33
5.2 An sample of input and outputs of different NLG system 34
5.3 Comparison of through human-based evaluation 36

List of Tables

5.1 Comparison through automatic metric evaluation 35

Chapter 1

Introduction

1.1 Natural Language Generation (NLG)

The Natural Language Generation (NLG) process actually wants to imitate the humans brain capability of speaking and writing in natural language by real-time (even faster than human brain). So, basically NLG is a part of artificial intelligence (AI) which is very common term nowadays. Alternatively, it can be seen that NLG is dealing with document generation in natural language. We know that the division of AI which concentrate on natural language tasks is named as Natural Language Processing (NLP). By this way, it is not tough to declare that NLG is a special sub-division of NLP. The main objective of an NLG system is to convert non-textual data (e.g. image, numerical data, graph, table, flowchart etc.) in to corresponding meaningful, fluent and easy understandable natural language text document. So basically an NLG system transforms obscure non-linguistic data to linguistically acceptable data.

1.2 Application Areas of NLG

There are huge application areas of NLG techniques in this present decade. Firstly an NLG system can be great equipment for a workspace where producing linguistic reports need every moment of time. Some example of such areas are news industries, real-time simulation systems, weather reporting and forecasting purpose, teaching environment, software reporting and many others. Before NLG come into research focus, those above mentions workspace were mainly used human support and previous database storages

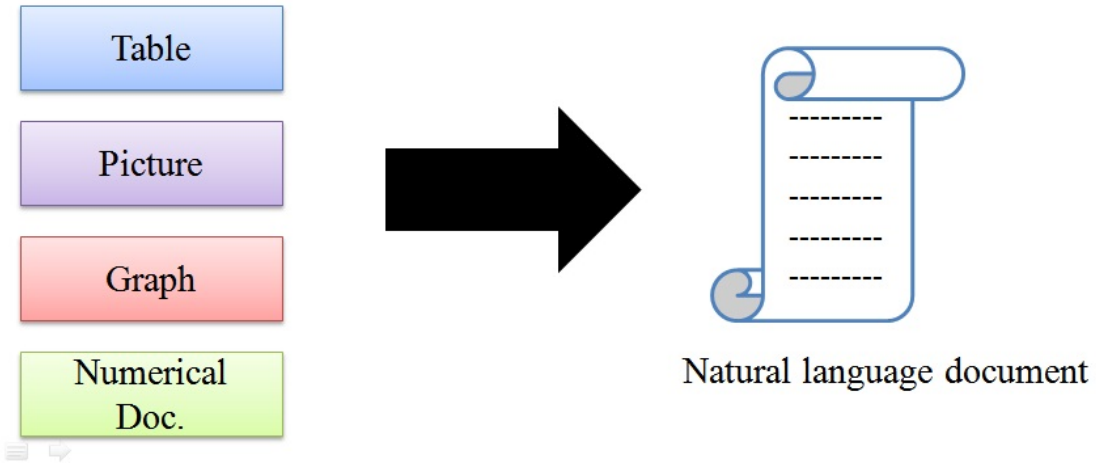

FIGURE 1.1: Tasks of an NLG system

to use the same report in the future. But those human techniques used to take too much time and also were quite expensive as there was a need to hired human writers as much as production required. Suddenly after the NLG concept eruption and their application in place of human stenographers, those workspace getting more and more cost efficient and also take very less time to generate report than a human writer can do. Some tremendous examples of application of NLG in current age are weather report generation from graphical map [1], representing medical information in a way so that patient could understand in easy way [2], to answer a question from a given knowledge base [3], automatic textual news generation from a short-script representation of reporter. There are many other application scope for NLG systems, one of them are in creating automatic recommendation system. In building a human interacted dialog system for any domain whether it game or gas station, NLG system provided a essential component of the dialog system. From the following picture 1.2 we can easily differentiate two

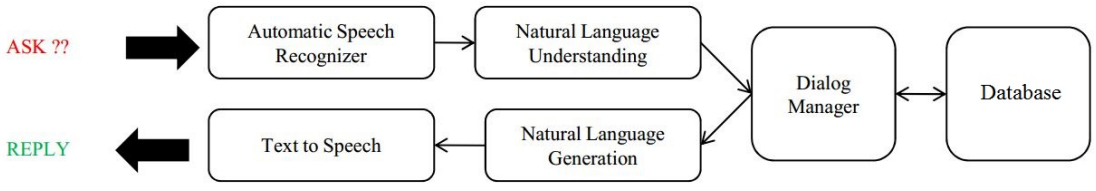

FIGURE 1.2: Application of NLG in Dialog System

confusing task of natural language processing, one is Natural Language Understanding (NLU) and another is NLG. Actually both are complement to each other from function strategies point of view. NLU actually extract out semantic non-textual representation from the textual document whereas NLG techniques do the just opposite work, it generate textual data from the underlying semantic or structure non-textual data. In dialog

system where it is for displaying or speaking, NLG component is must. The NLG component of the dialog system actually takes the dialog acts from the dialog manager and convert into it textual data format.

1.3 Where NLG Systems Are Not Usable

Some times it may happen that NLG system are not necessary for some application [4]. NLG system will not be appropriate where every time different types non-textual data need to be converted into corresponding textual document. In that case it would be better if agency hires a human writer instead of paying a large amount of money to NLG developers. Even in an application area where very high quality novel linguistic structures need to be produced every time for report generation, NLG system is also incapable of playing role in those application areas. So, there are few application fields where human writers outperform NLG systems but the number of such application fields is very low and most of the real world applications do not belong to those application areas.

1.4 Inefficient Template Based Techniques

Few alternative text generation techniques are also available in comparison to the NLG techniques. Among those alternative systems, template based generation systems are quite popular. In fact before NLG system invention, researchers mainly used template based system or mail-merge system for automatic report generation task. Although from theoretical perspective there is no difference between a complex mail-merge technique and a natural language generator, in terms of generated text quality and clarity [4]. But practically, one can observe some differences between template based generation systems and NLG systems. This is because, building a complex template based system is not a traceable task in any application domain as it takes a huge human effort and time. Some researchers prompted the difference between mail merge technique and the natural language generation technique, which are,

i. NLG system can give more syntactical language variation than a template based system can show.

ii. NLG systems are modular in structure and hence it is quite portable in nature. Shifting a NLG system from one application area to other doesnt require too much change in the existing NLG system. It only needs to change some modules of existing NLG system for new application domain field. But in case of template based system it needs to create almost the whole generation system for the new application domain areas. So NLG system has higher priority than a template based system with respect to this issue.

1.5 Approaches for Construction of NLG Systems

There are broadly two type of NLG system building approaches are present, first one is knowledge-intensive approach and second one is knowledge-light approach [5]. In knowledge-intensive approach the generation system takes lots of human and domain expert concept, effort and consultation for improvement of the system beside taking help from a parallel corpus. In one word knowledgeintensive approaches take many human help and data addition with the given parallel corpus data. In this approach if there are some wrong in parallel corpus it can be easily remove with a meeting to the domain expert of the corpus domain areas. The second approach are knowledge-light approach, compare to the knowledge-intensive approach it does not required any human or domain experts view on the given parallel corpus data. Knowledge-light generation systems are an actually fully statistical NLG system which doesnt require anything rather than a parallel corpus of the application domain areas. Knowledge-light approaches are mostly automatic in nature and use machine learning algorithms to build it from the parallel corpus. In contrast with the knowledge-light approaches, knowledge-intensive approaches are more like rule-based language generation systems. Although the knowledge-intensive approaches are rule-based in nature but most of the time the output quality of text from these approaches are quite good from human level evaluation [5].

1.6 Subtasks of an NLG System

Reiter and Dale [4] in their most influential NLG paper, had divided the whole natural language generation task into six almost disjoint subtasks. From many years after the paper came into light, most of the NLG researchers are believe that those six sub-tasks

of an NLG system are quite essential and exhaustive; hence most of them are consider those sub-tasks as a standard for any NLG system. According to Reiter and Dale [4], those six subtasks of an NLG are

i. Content Determination

ii. Discourse planning

iii. Sentence Aggregation

iv. Lexicalization

v. Referring Expression Generation

vi. Sentence realization

By the way, although there are six sub-task of the NLG are presents, but the task of an NLG is to convert a non-textual data into an understandable natural language textual data. It also be a valid NLG system, if any designer can assure to fulfill the task of NLG system without any consideration on the six subtasks of NLG system.

1.6.1 Content Determination

In content determination subtask, designer thinks about the central thought or expression which she/he going to express in natural language text. The central thought or expression or context is represented by some message or a packet of information [4]. A message can consist of several arguments or entity, a message identifier and a message relation. Messages relation is typically represented principal semantic orientation of that message. Messages entities are connected each other through the message relation. Message identifiers are actually used to distinguish different messages from message database. A NLG system can be expressed as a manipulator of a large collection of message. Remember one thing for every linguistic expression there will be unique message for a particular NLG system. Although message representation can be vary from one NLG system to other NLG systems.

1.6.2 Discourse Planning

In discourse planning, the appropriate sequence and association between multiple messages which is going to be part of output natural language text are mainly concerned [4]. It can be easily understand that any arbitrary sequences of messages will never express a fluent and semantic statement. There is particular sequence or order need to be imposed among collection of messages such that they can represent a real life thought and meaning. Association among the messages are represent the semantic collaboration between two different message and also the way in which they are linked together. Association between one than one simple message messages create a complex message. To express association between two or more messages discourse relation need to be incorporated among them. Discourse relation connect multiple messages each other with related semantic meaning. The output of discourse planning stage are treelike structure where root node stand for whole central thought of the conversation and leaf node of that tree represent simple messages which are part of the output natural language text. Like content determination subtask discourse planning is also a hefty subtask which required human effort and predefined knowledge about application areas of NLG system. Currently most of the NLG systems having discourse planning stage are mainly using rule base technique for it.

1.6.3 Sentence Aggregation

In this subtasks, possibility of merging multiple messages into a single sentence is been searched. Actually this subtask is not as essential as rest of the five subtask of an NLG system. But this subtask enhances readability and realization of the output generated text by alleviating very simple sentences and merging them into complex sentence [4]. There are some possible ways for sentence aggregation,

i. By conjunction
   Two sentence can be added with together by coordinating conjunction.

ii. By ellipsis
   Multiple sentences can be joined together by removing some recurring or repeated parts from those sentence.

1.7 Lexicalization

Till now above three subtasks only deal with the fact that how does look like syntactic structure of the output natural language text data. But this subtask, Lexicalization [6], is responsible for choosing appropriate vocabulary for each entities of message. From the context determination subtask, we learned that each entities of message represent a concept of the message. But context determination subtask does do select appropriate word or phrase for each concept. Lexicalization is one of the important subtasks of an NLG system, because it not only increases readability and clarity of output generated text but also responsible for carry out actual semantic values for each domain entities. Lexicalization subtask is also do selection for choosing word or phrases in place for messages and discourses relation.

There are numerous methods and rules are available for how to choose appropriate word for messages entities and relations from a collection of vocabulary set. One of most influential technique used for lexicalization purpose is graph rewriting technique [4], which is quite famous and has been applied in many application areas.

1.8 Referring Expression Generation

Actually referring expression generation does quite similar task of choosing appropriate word or phrase just like lexicalization subtask. Referring expression generation mainly further smooth words or phrases selection summary which resulted from lexicalization process. That means, even after lexicalization done its work, still there can be some ambiguities in the message, e.g. same naming object may be occurred in a message multiple times. Same named objects cannot create good surface sentences most of the times. That why referring expression generation plays a vital role here. Rather than choosing same object name, referring expression took some other form of the objects like pronoun or other name descriptors. Selecting appropriated referring expression for each domain or messages entities is not an easy-task its takes a lots observation on context of the output text.

For choosing most likely referring expression for a entity, designed should goes through three steps: Firstly initial introduction of the entity in text need to be accounted; Secondly for more than one occurrence choose appropriate pronoun for the entity; Thirdly in future reference incorporate definite descriptors for the same entity which has been

already occurred in preceding text. There are some referring expression generation techniques available into the research areas. Dale and Reiter [7] was proposed a technique for generating referring expression.

The output of lexicalization and referring expression generation subtask is tree-like structure same as output of the discourse planning subtask except in this tree-like structure all relations name and the leaf node entities are replaced by their appropriate reference vocabularies.

1.9 Sentence Realization

Sentence realization is one of important and last subtask of an NLG system among six earlier specified subtasks. This subtask sometimes also called as surface generation subtask. The objective of sentence realization subtask is take referring expression generation subtasks tree-like structure output and converts it into the real world linguistic sentence. Sentence realization subtask need to care about fluency and authenticity of output natural language text. This subtask should produce output text which are syntactically, semantically, morphologically and orthography correct. Syntactically correct means the output text must follow appropriate grammar rules and regulations. Semantically correct output text gives assurance of authenticity for the generated natural language text data. Through correct use of morphological form resulted text from sentence realization subtask are remained fluent and readable. Orthographically correct means put appropriate letter case in the out text; e.g. always start a sentence with uppercase letter and write any abbreviation word in all uppercase letters.

There are many available approaches are used for sentence realization subtask. There are some unique qualities among those each approaches. Some of them are fully rule based and other are automatic (fully and partial) generation approaches. One approach of sentence realization subtask is inversing of parsing [8]. In parsing system, parser creates a parse tree from a valid linguistic surface sentence. If we look at the sentence realization subtask, it can be easily seen that this subtask takes tree-like information from its preceding subtasks output and generate corresponding meaningful sentence from that intermediate tree-like structure. Hence the job of sentence realization subtask is exactly reverse job of a general parser, except parse tree and tree-like structure input of sentence realization are quite different. So by this concept, sentence realization subtask can be build from a Bi-directional grammar where the grammar can generate real world

sentences from their corresponding tree like structures [8]. Another approach of fulfilling sentence generation task is Meaning Text Theory [9]. It is one of the most popular sentence realization techniques which have been used in most of NLG system available by the date. MTT sentence realization takes deep syntactic tree structures of input data forms and turned those into corresponding textual sentential data. There is an another way to performing sentence realization subtask, with using Systematic Functional Linguistic (SFL) [10, 11]. SFL concerns with a collection of functions corresponding to a language and also have systematic grammar component which mapped those functions in to surface form of the language. This systematic grammar does the job of selecting appropriate functions for semantic results and keeps an order between them. Unlike other linguistic grammars in natural language, systematic grammars doesnt select appropriate grammar rules in every steps of its performance, rather it make choice of putting function from the fine-grain level to higher.

1.10 Architecture of NLG Systems

Over the years many architecture for NLG system has been proposed. The most important criteria in selecting an appropriate NLG system architecture are modular in designer, because modular design gives flexibility to modify and update existing NLG system through some more information and modification in future.

We will discuss the mostly accepted modular NLG architecture which based on the earlier mention six subtasks [12]. One thing needs to be noted here that, based on the six subtasks there are many modular architecture were evolved. One simple type of architecture, make module for each subtasks of NLG system. That means six modules will present in this architecture. But the problem in this type of architecture is with too many module existence and some of modules do the same objective.For example referring expression generation and lexicalization modules work on close to same objective with choosing appropriate word or phrases for entities. So there is no need to make two different modules for a single objective, it will be better if we merge them into one module. The most popular modular NLG architecture which is accepted by almost all NLG researchers, has three different modules and their objectives are totally different from one another. These three modules are given below,

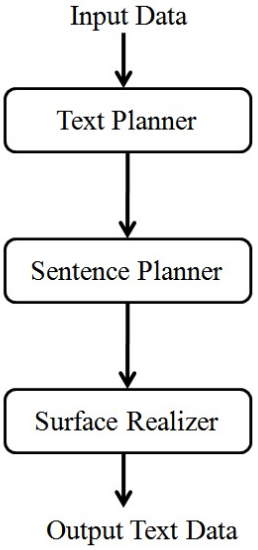

FIGURE 1.3: An architecture for NLG system

i. Text Planer

This module actually made of content determination and discourse planning subtask. Responsibility of a text planner is to choose appropriate structure and semantic focus for natural language text. The output of this module is tree-like structure of intermediated language. The output of text planner is named as text plan.

ii. Sentence Planner

This module do accumulated task of sentence aggregation, lexicalization and referring expression generation subtask. The main concern of this module is to generate right pre-platform for actual surface formed text output of NLG system. The output of sentence planner is also a tree like structure, but quite similar to surface natural language text rather than simplified tree-like structure output of text planner. The output of sentence planner is known as sentence plan.

iii. Surface realizer

This is the last module of the above mention NLG architecture. This module is stand for the sentence realization subtask. It takes tree structure output from sentence plans and generates real linguistic sentence form of that tree structure.

1.11 Problem with this NLG Architecture

Up to this point, we discussed the most acceptable and recognizable NLG systems description and its architecture. Basically the aforementioned NLG systems guideline leads us to the knowledge intensive natural language generation technique. But as we mention earlier that although knowledge intensive natural language generation system can generate good and grammatically correct sentence, they take huge human effort and most of the time become domain biased. In this type of NLG building strategy, sentence planner and text planner both module take lots of human effort and explicit understanding about the language, specially the text planner module required lots of helps from experts and designers.

Compared to these knowledge intensive approaches, knowledge light approaches take very less humans and domain experts helps or interactions. Knowledge light approaches mainly work with different type of available automatic machine learning algorithms and technique which dont take much of human effort. It is also need to say here that output text quality of knowledge light approaches are quite appreciable and understandable to anyone, although sometimes not as good as knowledge intensive approaches. Knowledge light approach is basically a statistical automatic NLG system building strategy. But only one thing that a knowledge light approaches are not assured that whether the built NLG system through these approaches are domain independent or not. If one can give assurance of domain independent properties in knowledge light approaches then it will be turned into a better and efficient approach.

1.12 Our Proposed NLG system

In this thesis work we have been proposed a novel knowledge light approach to build NLG system, which does not require any human helps or efforts and also domain independent in nature. Our proposed NLG system belong to the category of statistical supervised learning system. For learning or training purpose, the proposed system takes a parallel corpus as input, where non-textual tuple formed data and corresponding textual data are present.

We have made some assumption on dataset or corpus and the type of the dataset. Our proposed approach will performed its best if those assumptions are fulfilled. There are mainly two assumption are need to hold on the dataset,

i. Assumption 1
The non-textual data of corresponding to textual data of dataset must be tabular in nature. So that we can say that each non-textual data is represented by a tuple of the non-textual portion of dataset or corpus.

ii. Assumption 2
Most of attribute values or slot values of a non-textual data must be present into its corresponding textual data.

As we have mentioned earlier that our proposed approach is knowledge light approach, so it basically is a statistical tools and use mostly machine learning algorithms. In this approach we generate natural language text from the appropriate form of non-textual tuple data in basically two steps.

Step 1 We build a special type of graph (namely Attribute graph) to express the whole parallel dataset or corpus through that single graph. We have used that particular graph to efficiently perform prediction tasks on the given parallel dataset.

Step 2 After successfully creation of attribute graph, we basically perform some predictions on that graph to generate appropriate textual output for a given new tuple formed non-textual data. At the end we also consider linguistic quality of text and some issues regarding hefty search space of output text data.

1.13 Contribution

The following points are the main contributions which are part of this thesis:

i. We proposed a domain independent knowledge light approach based NLG system.

ii. We have shown an efficient structure of storing tabular form parallel dataset or corpus (where non-textual data are represented by a tuple).

iii. An extensive evaluation and comparison of the different existing NLG systems against our proposed system has been carried out to reflect the performance of our System.

1.14 Structure Of This Thesis

The thesis is structured as follows.

Chapter 1: This chapter mainly gives a broad overview of NLG systems and corresponding state of arts. In this chapter we also try to introduce our proposed system in brief manner.

Chapter 2: This chapter discusses mostly about state of arts NLG systems which are quite parallel to nature of our proposed NLG approach.

Chapter 3: This chapter gives a general overview and description of our task and also describe design strategy which lies behind our proposed system.

Chapter 4: This chapter elaborates the construction process of attribute graph from a parallel corpus and text generation process using attribute graph.

Chapter 5: This chapter describes the dataset which we have been used for evaluation purpose. In this chapter we also include the evaluation results of the proposed system and compared them against other NLG systems.

Chapter 6: This chapter concludes the thesis and provides avenues for future work.

Chapter 2

Related Works

In this chapter we introduce some work and research areas that are related to our topic of this thesis. Here we mainly present five type of techniques used in the NLG systems so far by today. We briefly mention the five types of NLG system building techniques.

2.1 Rule Based Techniques

This technique is also known as handcrafted technique for building NLG systems. In this technique, all the subtasks of an NLG system mentioned in the 1, used to be done though human helps and effort. All functional sections of an NLG system are depend on the rule which are written by some human designers manually in past. All the grammar rules of the system are also created manually. This type of techniques required too much human effort. These types of NLG building technique falls into the category of Knowledge Intensive approaches. This type of system are not only time consuming but also very domain and language dependent. Suppose, if one thinks to modify NLG system from an application domain to another different application domain then almost all part of the NLG system needs to be changed by manually, which is very cumbersome process. The same situation is also occurred in case of changing language of an NLG system building through rule-based techniques. But it need to be mentioned, although rule based techniques take too much cost, some of them generate quite good and exceptional natural language text most of the time. Following we gives some examples of rule base technique/systems.

2.1.1 SumTime System

SumTime System [6] is one of most popular NLG system among the all available NLG systems in this age . This is a rule-based system, which has two modules: a content-determination module and a sentence planner and realisation module. Without the content-determination module the SumTime system is known as SumTime-Hybrid system. SumTime system is also applied to the real life application. One notable thing of the SumTime system is that it output text qualities are fantastic and easily readable and understandable. But the problem that has in the SumTime system, is too much time and human effort need to be incorporated to build this system. The SumTime system designer took almost 12 person-months to build this NLG system and takes 24 person-months to get feedback from the weather forecasting experts to improve performance of the system [13].

2.1.2 FoG System

FoG was one of another popular rule based NLG system which discovered by Goldberg [1]. FoG is a bilingual system that generates English and French marine forecasts from a basic content representation, and was one of the first commercially used NLG systems.

2.1.3 PLANDOC System

PLANDOC [14] is an example of rule based handcrafted NLG system. It was build with the purpose of generating summary in a telephone network. The system takes simulation log as input, and produces a natural language text summary as description for the input simulation log record.

2.2 Handcrafted Grammar Based System

Any NLG system for a particular domain can be created with a set of grammar rules. Sometimes for building NLG system, this set of grammar rules are acquired by handcrafted way. The set of grammar rules must maintain the criteria that any possible sentences of a particular a NLG system can be generated through the set of rules.

2.2.1 Probabilistic Context Free Grammar Based NLG System

Here in this NLG system, context free grammar are used for natural language text generation purpose. But probability is associated with each context free grammar rule. Those probabilities used to determine the most possible sentence among many generated sentences through a set of grammar rules. There are many ways of constructing PCFG grammar. But Blez [15] discussed the Probabilistic Context-free Representationally Underspecified (PCRU) [13] language generation approach for constructing PCFG grammar. In this approach, a CFG is created manually through human observer that encompasses the whole set of all possible generation processes from inputs meaning representation to outputs natural language text, but this CFG has no decision-making ability. A probability distribution over this handcrafted CFG is calculated from a corpus, with this probability parameters the CFG acquired the decision making capabilities. Through the PCRU techniques one can build several different type of PCFG NLG systems: like Greedy-PCFG System, Viterbi PCFG system, Greedy roulette-wheel PCFG system etc.

2.3 Automatic Statistical Surface Realization System

In this type of NLG system, the text planner and sentence planner are remained as handcrafted module but the surface/sentence realizer become fully automatic. The grammar rules for sentence realisation are extracted directly from treebanks using statistical methods. The advantage of extracting the grammar rules automatically from treebank is that it reduces the manual effort in writing the rules. Another benefits of this technique is that the sentence realization module become domain independent and also language independent if a treebank of corresponding language are present in the past. We give an example of such system in below.

2.3.1 HPSG-based sentence realizer

In this type of generator uses Head-driven phrase structure grammar. But with HPSG most of the time ambiguities are aroused. So to alleviate those ambiguities, statistical methods are applied to the HPSG surface realization module. Those statistical methods keep the surface realization module as automatic.

2.4 Semi-automatically Extracted Rule Based Generator

This type of sentence generator uses grammar rules which are made through both handcrafted and automatic statistical policy. Most of the time, designer picks the grammar rules through handcrafted techniques and apply those manually chosen grammar rules probabilistically in sentence generation. Automatic nature of sentence generator is mainly concern with choosing best generated sentence among several possible sentences which can be generated.

2.4.1 Probabilistic Synchronous Context-Free Grammar Based Generator

This type of language generation system is based on synchronous context-free grammars, which are mostly used in machine translator. In synchronous context-free grammar, a pair of grammar is used. One grammar in that pair is responsible for generation of meaning representation for the input data and the other grammar of the pair is responsible for generating natural language text for corresponding input data. For each rule of a meaning representation grammar there will be at least one rule for the output natural language text generation grammar. Belz [13] used the $WASP_{-1}$ method [16] to generate the PSCFG grammar almost automatically from a parallel corpus where input representation and corresponding natural language text co-exist. The training process of PSCFG grammar in this $WASP_{-1}$ method is gone though two phrases, one is acquisition phrase and another is parameter estimation rule. In acquisition phrase, lexical rule construction is concerned. The parameter estimation phrase associate probability as a parameter to those rule which are discovered in the acquisition phrase.

2.5 Fully Automatic Language Generation Systems

This type of NLG system are fully automatic and does not need any human effort or handcrafted knowledge in construction unlike above mentioned four type of NLG system. In the most of cases of automatic language generator become statistical supervised system. Till date, a lots of knowledge-light approached based automatic language generator system has been proposed and also achieved quite good results in generation task.

Few of earlier statistical natural language generation systems were Nitrogen [17, 18] and Oxygen [19]. These two NLG system were based on statistical sentence realizer.

2.5.1 Mountain System

Recently Mountain system [20] has been designed, which used a concept of machine translation into the natural language generation process. This Mountain generation system takes helps from MOSES[1] toolkit [21] for the machine translation task.

2.5.2 Generation Through Phrase-based Statistical Machine Translation

Another statistical machine translation based NLG was present in Phrase-based statistical machine translation (PB-SMT) [15] model. PB-SMT based NLG systems are belong to statistical knowledge light approached based language generation system. The MOSES toolkit offers an efficient way for building the PB-SMT model. However, the linguistic quality and readability of PB-SMT based NLG systems was not good compared to relevant statistical NLG system.

2.5.3 Case Based Natural Language Generation

Another type of automatic NLG systems made through case-based reasoning or instance based learning. This category of CBR based NLG system based on the concept that similar set of problem will appear in future and same set of solution will compensate or solve for that problems. SEGUE NLG system [22] was partly CBR based NLG system, more accurately SEGUE had been made with mix of CBR approach and rule based policy. In article [5], a CBR approach based weather forecasting text generation tool, CBR-METEO has been designed. The advantage of CBR based system was it takes very little manual helps and if given prior dataset contains almost all type of input instances then CBR based system also performs better.

[1] http://www.statmt.org/moses

2.5.4 Neural Network Based Automatic Language Generator

Recently, some neural network based NLG systems has been proposed. With the advent of recurrent neural based language model (rnnlm) [23] many rnn based NLG system has been proposed. An idea of generating text through recurrent neural network based approach with Hessian free optimization has been proposed [24]. But this method used to take a large training time. An influential rnn based NLG techniques has been proposed on [25] which based on a joint recurrent and convolutional neural network structure. That system has able to get trained on dialogue act-utterance pairs without any semantic alignments or predefined grammar trees.

Chapter 3

Problem Overview

As in chapter 1, we have mention that The task of natural language generation (NLG) is to generate textual data from any form of non-textual data which can graphics, figure, raw numerical databased etc. This our project we mainly concentrated on tuple formed slot-value pair non-textual data.

We proposed a knowledge-light approach based, statistical, supervised NLG system for generating natural language text data from a non-textual tuple formed data. Like any other computer software system, the proposed system goes through similar development stages. Primarily those major development stages are task definition, requirement analysis, designing strategy, development and maintenance. For our case we dont show the maintenance stage into this thesis as we are not concern with that stage. In this chapter we have considered only the task definition, requirement analysis and designing strategy stage. In next chapter we have shown the development phase of proposed NLG system.

3.1 Task Definition

The principle objective of the proposed system is to generate natural language text output from tuple record of a non-textual table structure dataset. The table contains a set of different attributes (qualitative or quantitative). Each row or tuple of the table represents a single unit of non-textual data which represents a vector of that particular tuples attribute values. Figure 3.1 visualizes this task. According to that figure, the task is to generate textual data t_{tx} from a tuple-formed non-textual data t_{ntx} of the T_{ntx} dataset

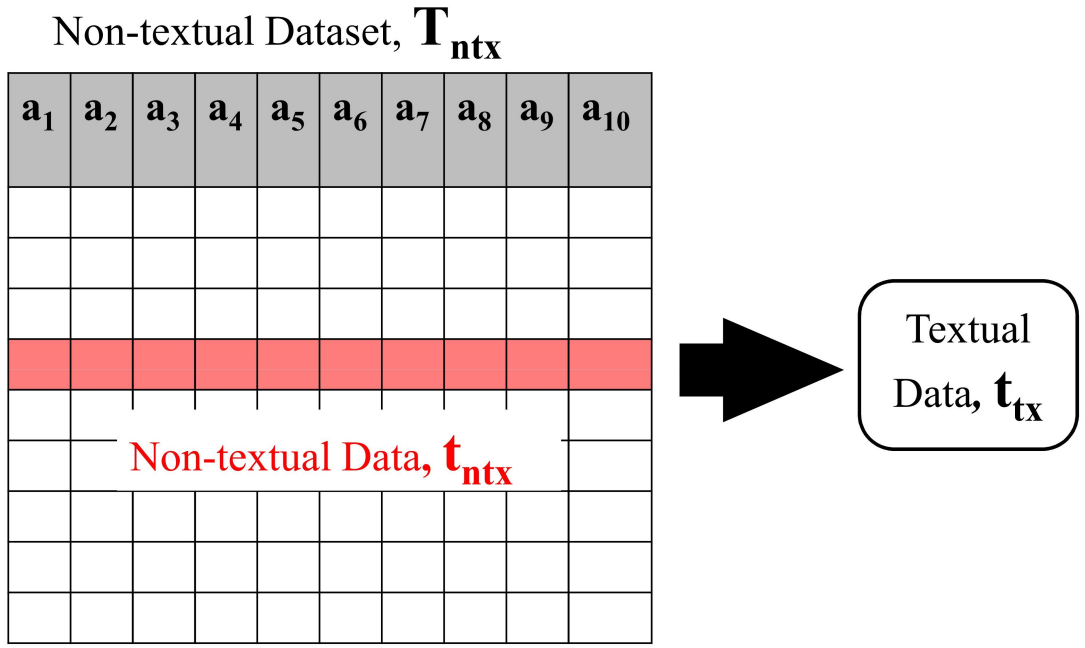

FIGURE 3.1: Basic input-output structure of our system

(table). The T_{ntx} dataset (table) has ten attributes, a_x, where x = 1....10; however, the actual attribute names and attribute values are not shown in the figure.

3.2 Requirement Analysis

Being a supervised statistical model, the proposed system needs a parallel corpus for training purpose which should be a collection of non-textual tuple data and the corresponding textual data. We make an assumption here, that most of the attribute values present in any non-textual data should also appear in the corresponding textual data for that non-textual data. For our experiments we used the Prodigy METEO[1] [**?**]

3.3 Design Strategy

To generate human readable and easily understandable textual data from non-textual data, an NLG system must ensure two criteria. Firstly, output natural language text should be related to the corresponding non-textual datas topic, i.e., output text must

[1]http://www.nltg.brighton.ac.uk/home/Anja.Belz/Prodigy

contain appropriate information. Secondly, the output text must be fluent, i.e., the output text must ensure its Linguistic quality.

Step 1: Non-textual data t_{ntx}

rainfall	temp (morning)	temp (afternoon)	temp (night)
no	13	30	10

Step 2(iterative):

I. Attributes' name sequence $s_{t_{ntx}}$ for t_{ntx} (note: no match in found on 'rainfall' attribute value)

<s> temp(morning) temp(afternoon) temp(night) </s>

II. Interlinked word-group $iw_{x-y}^{t_{ntx}}$

sharply increase to

by midnight with zero rainfall

Temperature

by afternoon and decrease to

Step 3: Result of step 1 and step 2, t_{tx}

<s> Temperature 13 sharply increase to 30 by afternoon and decrease to 10 by midnight with zero rainfall </s>

FIGURE 3.2: Designing strategy behind our proposed model

This concept is illustrated in figure 3.2. It shows the steps which are necessary for maintaining informativeness of the output textual data. The example shown in the figure 3.2 is taken from a non-textual tuple formed data from a temperature and rainfall weather dataset which is not our actual experimental dataset; it is presented only for illustration purpose.

Chapter 4

Development Phase (Text Generation)

This chapter is dedicated to the development phase of our proposed NLG system. The chapter is consisted of two parts. In the first part we concern with building of Attribute Graph from a given parallel corpus, where non-textual data and textual data are coexist. The second part elaborate how we can generate natural language text from the built attribute graph.

4.1 Attribute Graph

4.1.1 Intuition of Attribute Graph

The NLG technique presented in this thesis is focused around Attribute-Graph. The notion of Attribute-Graph partly resembles with word lattice [17] which gives an idea for natural language generation without incorporating any hard content determination effort on training knowledge base. A word lattice is defined as a directed acyclic graph with one starting and final state. Each edge in word lattice is labeled with a word from the given knowledge base. A walk from the starting state (or vertex, or node) to the final state results in one of the possible text outputs from the word lattice. Vertices having more than one outdegree are sources for multiple text outputs. However, all random walks on the word lattice do not produce real world text as output because of too much freedom in the random walk and existing ambiguity in the word lattice.Figure 4.1 shows a word lattice built from three sentences. It also shows two example sentences generated through random walks in this word lattice, one of which is semantically correct while

the other is not. We adapt the idea of word lattice to our text generation framework

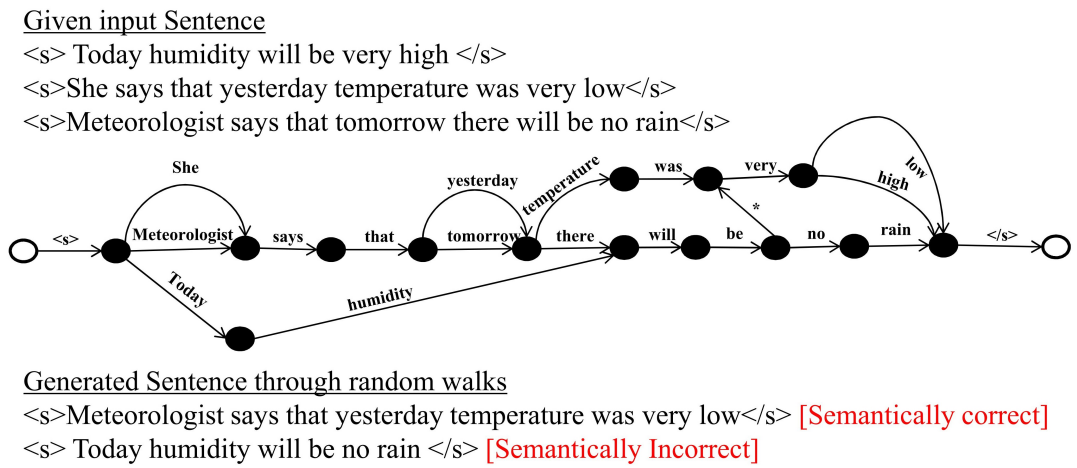

FIGURE 4.1: Word lattice formation and generation of text

by introducing two important modifications in word lattice to arrive at the concept of attribute graph. The first modification is that vertices's in an attribute graph represent words corresponding to values of an attribute from the given tuple formed non-textual data. The second change is that an edge in an attribute graph can represent not just a word, but a sequence of words as well. We noticed that it is possible to generate good quality text by making predictions along an 'appropriate' path in the attribute graph from the starting state to the final state. The notion of an appropriate path comes from answering the question - "Which attributes of a tuple-formed non-textual data will be present in the corresponding generated textual data and what will be the order of those attributes in the generated textual data?". In the work presented in this paper we predicted the answer of that question by observing the training dataset. Surprisingly, after ensuring this prediction we found that the output of our NLG system becomes semantically more relevant to its corresponding non-textual tuple-formed data.

4.1.2 Building of Attribute Graph

The concept of attribute graph draws from and resembles with both word lattice and multipartite graph. The attribute graph $G_A(V, E)$ is a directed multipartite weighted graph built from a parallel dataset (corpus) where textual data $(t^i_{tx} \in T_{tx})$ and corresponding non-textual (tuple formed) data $(t^i_{ntx} \in T_{ntx})$ coexist. In this discussion we use the term 'data' $(t^*_{tx/ntx})$ to represent each row in $(T_{tx/ntx})$ 'dataset' ,where'*' is a number placeholder. For example, t^1_{tx} represents the textual data contained in the first

row in the T_{tx} dataset. a_x corresponds to the x^{th} attribute name in T_{ntx} and $a_x^{t_{ntx}^1}$ represents the slot value of a_x attribute for the t_{ntx}^1 data.

Each partition set $(pset_*)$ in an attribute graph corresponds to an attribute name (a_*) from the non-textual dataset T_{ntx} in the training data. For example, if a given non-textual tuple formed data consists of the attributes: {temperature, wind-speed, humidity}, then 'temperature', 'wind-speed' and 'humidity' will all be represented as different partition sets in the attribute graph for that dataset. All the values of the 'temperature' attribute in the T_{ntx} dataset create the partition set $pset_{temparature}$ in the attribute graph. A vertex in the attribute graph represents a value of an attribute of T_{ntx} suggested by the corresponding partition set in which the vertex belongs.

For creating the edges $(e_* \in E)$ in an attribute graph during the construction phase, we have made a tricky assumption that some of the attribute values of a non-textual data (t_{ntx}^i) will appear in the corresponding textual data (t_{tx}^i).

An edge between two vertices in an attribute graph for a pair of non-textual (t_{ntx}^k) and textual (t_{tx}^k) data represents interlinked words between two aligned adjacent vertices' named attributes' values in the textual data (t_{tx}^k).

Algorithm 1 presents the procedure for building the attribute graph from a given parallel data-text dataset. Figure 4.2 shows a sample attribute graph built from two textual data (say, t_{tx}^i and t_{tx}^j) along with their corresponding non-textual tuple-formed data (t_{ntx}^i and t_{ntx}^j) taken from the training set.

4.2 Text Generation Using Attribute Graph

To generate human readable and easily understandable textual data from non-textual data, an NLG system must ensure two criteria. Firstly, output natural language text should be related to the corresponding non-textual data's topic, i.e., output text must contain appropriate information. Secondly,the output text must be fluent, i.e., the output text must ensure its Linguistic quality.

In this generation phase, we divide our proposed system into two modules; one module holds the responsibility of ensuring informativeness and the other module maintains linguistic quality of the generated output text.

Input: Parallel data-text dataset (T) {textual data $(t^i_{tx} \in T_{tx})$ and non-textual tuple formed data $(t^i_{ntx} \in T_{ntx})$ and $T = T_{tx} \cup T_{ntx}$}
Output: Attribute-Graph $G_A(V, E)$

Initially, create two vertices for endpoints of all textual data, $v_{<s>} \in V$ and $v_{</s>} \in V$ along with the corresponding values, $\{< s >\}$ and $\{< /s >\}$.
for *each non-textual data* $(t^i_{ntx} \in T_{ntx})$ **do**
  for *each attribute* a_x *in* t^i_{ntx} **do**
    Create partition $pset_{a_x}$
  end
  for *each attribute value* $a_p^{t^i_{ntx}}$ *in* t^i_{ntx} **do**
    Create a vertex $v^i_{a_p} \in V$ in partition $pset_{a_p}$
    if *the attribute value* $a_p^{t^i_{ntx}}$ *appears in the corresponding textual data* t^i_{tx} **then**
      Replace the attribute value $a_p^{t^i_{ntx}}$ with the attribute name a_p in t^i_{tx}
    end
  end
  for *for each attribute value* $a_p^{t^i_{ntx}}$ *and* $a_n^{t^i_{ntx}}$ *of* t^i_{ntx} *which are replaced in* t^i_{tx} **do**
    Find out the attributes corresponding vertex $v^i_{a_p}$ and $v^i_{a_n}$
    if *if* a_p *and* a_n *are adjacent (have no other* a_* *in between) in* t^i_{tx} **then**
      Insert a edge $e(v^i_{a_p}, v^i_{a_n}) \in E$ from vertex $v^i_{a_n}$ to $v^i_{a_n}$, direction of the edge will be same as occurring order of a_p and a_n in t^i_{tx}
      Find all the interlinked words $iw^{t^i_{ntx}}_{p-n)}$ between the two endpoints words a_p and a_n in t^i_{tx} and assign $iw^{t^i_{ntx}}_{p-n)}$ as weight for the edge.
    end
  end
end

Algorithm 1: Build Attribute Graph

4.2.1 Informativeness Management Module

We define informative quality of a generated output text by considering how many attribute values of its corresponding non-textual data are present in the generated output text and in which order. It can be noted that if a given parallel corpus holds our requirement analysis criteria mentioned in Section 3.2 then we can represent each textual data as a sequence of attribute-names (later should be replaced with attribute-values) of its corresponding non-textual data (which is also partition sets' name sequence) with interlinked word-groups between two adjacent attribute values present in that sequence.

So basically We subdivide the informativeness module into two submodules. First submodule is responsible for predicting the appropriate sequence of partition sets' name

Non-Textual Tuple Formed Data

t^0_{ntx}	Temperature (at morning)	Temperature (at afternoon)	Temperature (at night)	Rainfall (cm)
t^i_{ntx}	13	30	10	No
t^j_{ntx}	14	21	9	13

Corresponding Text Data

t^i_{tx} <s> Temperature 13 sharply increase to 30 by afternoon and decrease to 10 by midnight , there is no rainfall reported in this day </s>

t^j_{tx} <s> Temperature 14 slightly rise to 21 by afternoon again decrease to 9 later in the night, there is a report of 13 cm rainfall till midnight </s>

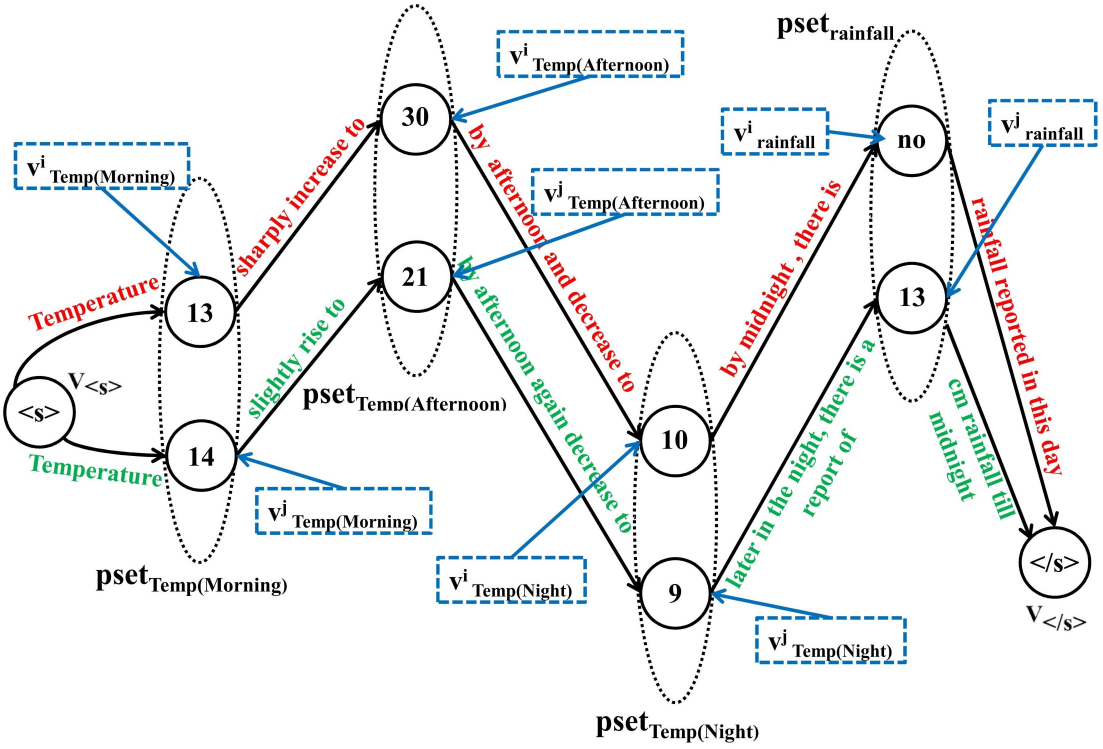

FIGURE 4.2: Attribute Graph

while the second submodule's job is to select all the interlinked word-groups that should be present in the partition sets' name sequence predicted by the first submodule.

Let assume we going convert a new test non-textual data t^{new}_{ntx} in to corresponding textual data t^{new}_{tx}. To achieve this we first need to find out the partition sets' name sequence $pseq_{t^{new}_{ntx}}$ and thereafter identify all interlinked word-groups $iw^{t^{new}_{ntx}}_{*-*}$. Mathematically we

can express this as in Equation 4.1 since t_{tx}^{new} is made up of $pseq_{t_{ntx}^{new}}$ an $iw_{*-*}^{t_{ntx}^{new}}$.

$$P(t_{tx}^{new}|t_{ntx}^{new}) = P(pseq_{t_{ntx}^{new}}, iw_{*-*}^{t_{ntx}^{new}}|t_{ntx}^{new}) \tag{4.1}$$

By applying probability chain rule we can write the equation 4.1 as below,

$$P(t_{tx}^{new}|t_{ntx}^{new}) = P(pseq_{t_{ntx}^{new}}|t_{ntx}^{new}) * P(iw_{*-*}^{t_{ntx}^{new}}|t_{ntx}^{new}, pseq_{t_{ntx}^{new}}) \tag{4.2}$$

Equation 4.2 rewrites Equation 4.1 as the product of two individual models where the first model, $P(s_{t_{ntx}^{new}}|t_{ntx}^{new})$, denotes prediction of partition sets' name sequence $s_{t_{ntx}^{new}}$ for the non-textual data t_{ntx}^{new}, while the second model, $P(iw_{*-*}^{t_{ntx}^{new}}|t_{ntx}^{new}, pseq_{t_{ntx}^{new}})$, denotes prediction of all interlinked word-groups $iw_{*-*}^{t_{ntx}^{new}}$ for t_{ntx}^{new} in the partition sets' name sequence $s_{t_{ntx}^{new}}$ predicted by the first model.

i. **Predicting partition sets' Name Sequence**

This model predicts the probable partition sets' name sequence for a given tuple-formed non-textual data. For this prediction, firstly each attribute value of a non-textual data needs to be identified in the corresponding textual data. It may so happen that some of the attribute values of the non-textual data might not be present in the corresponding textual data.But we must add those attribute values(as features) in predicting partition sets' name sequence, because they can provide some significant part in that predictions.

Let us consider that we want to find the partition sets' name sequence $pseq_{t_{ntx}^{new}}$ corresponding to some non-textual data t_{ntx}^{new}. Let the non-textual dataset T_{ntx} ($t_{ntx}^{new} \in T_{ntx}^{new}$) contains $a_1, a_2, ...a_n$ attributes and the corresponding attribute values in t_{ntx}^{new} are $a_1^{t_{ntx}^{new}}, a_2^{t_{ntx}^{new}}, ...a_n^{t_{ntx}^{new}}$. Then we can express the partition sets' name sequence prediction for t_{ntx} as given in Equation 4.3.

$$P(pseq_{t_{ntx}^{new}}|t_{ntx}^{new}) = P(pseq_{t_{ntx}^{new}}|a_1^{t_{ntx}^{new}}, a_2^{t_{ntx}^{new}}, ..., a_n^{t_{ntx}^{new}}) \tag{4.3}$$

After predicting the partition sets' name sequence for a test non-textual data, we replace the partition sets' names with their corresponding attribute names. So,at the end of this prediction we will get attribute name sequence $s_{t_{ntx}^{new}}$ for given a non-textual data t_{ntx}^{new}.

ii. **Predicting interlinked word-groups**

For a sequence containing n possible attribute names/values corresponding to a

textual data, there will be $n+1$ number of interlinked word(s) (or word-groups) as we introduce two default pseudo-attributes, one at the start and the other at the end of the text.

We predict the interlinking word(-group) between two attribute names along the attribute name sequence predicted in the earlier step for a test textual data from a context window of six attribute names around the two attribute names currently being considered. We also consider the attribute names of the non-textual data which do not appear in the attribute name sequence. Therefore, predicting the interlinked word-groups for a textual data are independent of each other. Let us consider that we want to predict the interlinking word-groups $iw_{*-*}^{t_{ntx}^{new}}$ for an attribute name sequence $s_{t_{ntx}^{new}}$ of a non-textual data t_{ntx}^{new}. Let this $s_{t_{ntx}^{new}}$ sequence be $[...a_l\_\_a_m\_\_a_n\_\_a_o\_\_a_r\_\_a_s\_\_a_t\_\_a_u...]$ and we want to determine the intermediate word-group between a_o and a_r. Let us also assume that some of the attributes' (a_e, a_f, a_g) values of t_{ntx} are not present in $s_{t_{ntx}}$, which are $a_e^{t_{ntx}^{new}}$, $a_f^{t_{ntx}^{new}}$ and $a_g^{t_{ntx}^{new}}$. We model this task of predicting interlinking word-group between a_o and a_r for t_{ntx} and $s_{t_{ntx}}$ as in Equation 4.4.

$$P(iw_{o-r}^{t_{ntx}^{new}}|t_{ntx}^{new}, s_{t_{ntx}^{new}}) = P(iw_{o-r}^{t_{ntx}^{new}}|a_m^{t_{ntx}^{new}}, a_n^{t_{ntx}^{new}}, a_o^{t_{ntx}^{new}}, a_r^{t_{ntx}^{new}}, a_s^{t_{ntx}^{new}}, a_t^{t_{ntx}}, a_e^{t_{ntx}}, a_f^{t_{ntx}}, a_g^{t_{ntx}}) \tag{4.4}$$

More precisely we can write the $P(iw_{*-*}^{t_{ntx}^{new}}|t_{ntx}^{new}, s_{t_{ntx}^{new}})$ term as the product of independent interlinking word-group prediction tasks as in Equation 4.5, where $prev$ (previous) and $next$ (next) are any of two adjacent attribute names in the $s_{t_{ntx}^{new}}$.

$$P(iw_{*-*}^{t_{ntx}^{new}}|t_{ntx}^{new}, s_{t_{ntx}^{new}}) = \prod P(iw_{(prev)-(next)}^{t_{ntx}^{new}}|t_{ntx}^{new}, s_{t_{ntx}^{new}}) \tag{4.5}$$

Therefore, Equation 4.2 can be rewritten as in Equation 4.6.

$$P(t_{tx}^{new}|t_{ntx}^{new}) = P(pseq_{t_{ntx}^{new}}, iw_{*-*}^{t_{ntx}^{new}}|t_{ntx}^{new}) \tag{4.6}$$

From the earlier discussion it have been understood that, $pseq_{t_{ntx}^{new}}$ and $s_{t_{ntx}^{new}}$ are both same things, we just replace the partition set names with their corresponding attribute names. So equation 4.6 can be write as equation 4.7.

$$P(t_{tx}^{new}|t_{ntx}^{new}) = P(pseq_{t_{ntx}^{new}}|t_{ntx}^{new}) * P(iw_{*-*}^{t_{ntx}^{new}}|t_{ntx}^{new}, s_{t_{ntx}^{new}}) \tag{4.7}$$

From equation 4.5 and 4.7 we can reduce the following equation 4.8.

$$P(t_{tx}^{new}|t_{ntx}^{new}) = P(pseq_{t_{ntx}^{new}}|t_{ntx}^{new}) * \prod P(iw_{(prev)-(next)}^{t_{ntx}^{new}}|t_{ntx}^{new}, s_{t_{ntx}^{new}}) \quad (4.8)$$

4.2.2 Linguistic Quality Management Module

The informativeness management module tries to answer "what will be content within the output textual data ", but it does not concern the linguistic quality, e.g., fluency, readability, etc. In the linguistic quality management module we try to deal with the deficiency of linguistic quality in the generated textual data. Statistical language modeling is a well established technique for ensuring fluency and readability in natural language text. Therefore, as a final component, we incorporate a language model in our system. Hence, to maintain both informativeness and linguistic quality of the generated textual data, we model the task of NLG as given in Equation 4.8, where $PP(x)$ stands for the perplexity of string x.

$$P(t_{tx}^{new}|t_{ntx}^{new}) = P(pseq_{t_{ntx}^{new}}|t_{ntx}^{new}) * \prod P(iw_{(prev)-(next)}^{t_{ntx}^{new}}|t_{ntx}^{new}, s_{t_{ntx}^{new}}) * PP^{-1}(t_{tx}^{new}) \quad (4.9)$$

For our experiments, we trained a trigram language model on the training set textual data with a minor modification by replacing the attribute values with their attribute names.

4.2.3 Decoding

The search space of the NLG problem as modelled by Equation 4.9 is enormous. For example, if we consider top ten attribute name sequences, and for each attribute name sequence there are overall fifteen interlinked word-groups and for selecting each of these interlinked word-group we consider only top five candidates, then the search space for the generation task will contain $10 * 5^{15}$ candidates. To reduce the size of the search space and keep the computation problem tractable, we implemented the task of text generation as modeled in Equation 4.9 using *stack decoding* [26] with histogram pruning which limits the number of most promising hypotheses to be explored by the size of the stack. In stack decoding with histogram pruning, the stack at any point of time contains

only N (size of the stack) most promising partial hypotheses and during hypothesis expansion, a partial hypothesis is placed on the stack provided there is space in the stack, or, it is more promising than at least one of the partial hypotheses already stored in the stack. In case of stack overflow, the least promising hypothesis is discarded.

Chapter 5

Dataset and Evaluation

This section presents the dataset used in our experiments and the evaluation results of proposed system compared to some other NLG systems.

5.1 Dataset

As mentioned in Section 3.2, to train our system a non-textual – textual parallel dataset is required. The parallelism should be in such a form that each non-textual data can be represented as a tuple of attribute value instances and most of those attribute values should be present in its corresponding textual data.

We used the Prodigy-METEO[1] corpus [27], a wind forecast dataset, for our experiment. In the Prodigy-METEO corpus a single pair of non-textual–textual data stands for a particular day's wind forecast report. A non-textual data in that dataset is represented by a seven-component vector, where each component expresses a particular feature of wind data measurement at a moment of time. The seven components belong to a vector represented by [*id, direction, speed_min, speed_max, gust-speed_min, gust-speed_max, time*]. In that vector representation *id* stands for identification of the vector, *direction* mentions wind speed direction, *speed_max and speed_min* denote maximum and mnimum wind speed respectively, *gust_max and gust_min* represent maximum and minimum wind gust speed respectively, and the last component *time* denotes the specific time instance when

[1] http://www.nltg.brighton.ac.uk/home/Anja.Belz/Prodigy

rest of components' readings were measured. For example, 1st April, 2001 wind forecast data is represented in this dataset as [[1,_SW,28,32,42,-,0600],[2,-,20,25,-,-,0000]] , where '-' represents a missing reading value.

As mentioned earlier our proposed model can process only a single tuple formed non-textual data at a time. However, the Prodigy-METEO corpus represents each non-textual data (wind forecast data for a particular day) by a sequence of multi-component vectors. For this reason, we merge all the vectors of a particular day's wind forecast data into a single tuple formed data. The merging of a a particular day's wind forecast data vectors is illustrated in Figure 5.1. In the Prodigy-METEO corpus, there are almost 490 non-textual tuple data are present with their corresponding textual data.

Corpus Non-Textual Data

Id	Dir	Speed_ max	Speed_ min	Gust_ max	Gust_ min	Ts
1	_S	20	24	-	-	0600
2	-	34	38	46	-	1800

Corresponding Textual Data
S'LY 20-24 INCREASING 34-38 GUSTS 46 BY EVENING

Our format of the Non-Textual data

Id(1)	Dir(1)	Spee	Spee	Gust	Gust	Ts(1)	Id(2)	Dir(2)	Spee	Spee	Gust	Gust	Ts(2)
1	_S	20	24	-	-	0600	2	-	34	38	46	-	1800

FIGURE 5.1: Transformation of Prodigy-METEO corpus non-textual data representation for our proposed system's input

5.2 Evaluation

We evaluated our system using both automatic evaluation metrics and human evaluation. For both human and automatic evaluation, we compared our proposed system with ten existing NLG systems whose outputs on the Prodigy-METEO testset are also available in the Prodigy-METEO corpus. These ten NLG systems are PCFG-Greedy, PSCFG-Semantic, PSCFG-Unstructured, PCFG-Viterbii, PCFG-2gram, PCFG-Roulette, PBSMT-Unstructured, SumTime-Hybrid, PBSMT-Structure and PCFG-Random [15]. Figure

shows a sample input and outputs of all the above mentioned systems including our proposed system.

Non-Textual Data	[[1,_SW-WSW,08,12,-,-,0600],[2,_VAR,02,06,-,-,1800]]
Corpus	SW-WSW 08-12 FALLING VARIABLE 02-06 BY LATE AFTERNOON
SumTime	SW-WSW 10 OR LESS BECOMING VARIABLE BY LATE AFTERNOON
PSCFG-Semantic	SW-WSW 08-12 FALLING VARIABLE 02-06 BY EVENING
PSCFG-Unstructured	SW-WSW 08-12 FALLING VARIABLE 02-06 BY EVENING
Our Proposed System	**SW-WSW 08-12 FALLING VARIABLE 02-06 BY EVENING**
PCFG-Greedy	SW-WSW 8-12 THEN FALLING VARIABLE 2-6 BY EVENING
PCFG-Viterbi	SW-WSW 8-12 THEN FALLING VARIABLE 2-6
PCFG-Roulette	SW-WSW 8-12 FALLING VARIABLE 2-6
PCFG-2gram	SW-WSW 8-12 THEN VARIABLE 2-6
PCFG-Random	SOON SW-WSW 8-12 THEN STEADILY EASING TO VARIABLE'LY 2-6 AHEAD OF THE FRONT FOR A TIME THIS AFTERNOON
PBSMT-Unstructured	LESS SW-WSW 08-12 GRADUALLY FALLING VARIABLE 02-06 BY EVENING
PBSMT-Structured	GUSTS SW-WSW 08-12 BY IN TO AND FALLING TO UNKNOWN VARIABLE 02-06 BY EVENING

FIGURE 5.2: An sample of input and outputs of different NLG system

5.2.1 Automatic Evaluation

For automatic evaluation, we used two automatic evaluation metrics; BLEU [28] and METEOR 29. Both BLEU and METEOR were originally proposed for evaluation of machine translation (MT) systems. However, due to the similarity between the two tasks (i.e., MT and NLG) from the point of view of their working principles, most of the NLG systems are also evaluated using these two automatic MT evaluation metrics.

BLEU metrics is particularly an n-gram based (modified) precision metric which measures similarity between two sentences (hypothesis and reference) by comparing their n-gram matches. It also employs a brevity penalty to penalize hypotheses that are shorter than their corresponding references. METEOR calculates unigram overlaps between hypothesis and reference and it uses a reordering penalty on how many chunks in the hypothesis need to be moved around to get the reference text. In addition to unigram matching, METEOR also considers synonym and stem matching.

System	BLEU score	Meteor score
Corpus	1	1
PCFG-Greedy	0.65	0.85
PSCFG-Semantic	0.64	0.83
PSCFG-Unstructured	0.62	0.81
Proposed System	**0.61**	**0.82**
PCFG-Viterbii	0.57	0.76
PCFG-2gram	0.56	0.76
PCFG-Roulette	0.52	0.76
PBSMT-Unstructured	0.51	0.81
SumTime-Hybrid	0.46	0.67
PBSMT-Structure	0.34	0.59
PCFG-Random	0.28	0.52

TABLE 5.1: Comparison through automatic metric evaluation

We experimented and evaluted our system on the predefined 5 splits of the dataset(Prodigy-METEO) in a cross-validation framework.

Table 5.1 presents the evaluation results obtained on our proposed system along with the ten other NLG systems BLEU and METEOR.

5.2.2 Human Evaluation

Evaluation using automatic evaluation metrics is very popular among researchers and developers since automatic evaluation is very fast and cheap. Automatic evaluation metrics are good indicators of system performance and they greatly help day-to-day system development. However, despite being very time intensive and costly, human evaluation still serves as the de-facto standard and the worth of automatic evaluation metrics are typically judged based on how well they correlate with human evaluation.

We also evaluated the systems using human evaluation on a part of test dataset. We carried out human evaluation to measure the clarity and readability of the texts generated by the NLG systems. Clarity measures truthfulness and fairness of a textual data whereas readability concerns fluency of textual data. 30 instances (out of total 232) were randomly chosen from the testset for the pilot human evaluation and the output from 11 different systems along with the corresponding non-textual data were presented to the human evaluators. Five students from different backgrounds who acted as human evaluators were asked to rate 72 outputs each in a 10 point scale.

The output of human evaluation is presented in Figure 5.3.

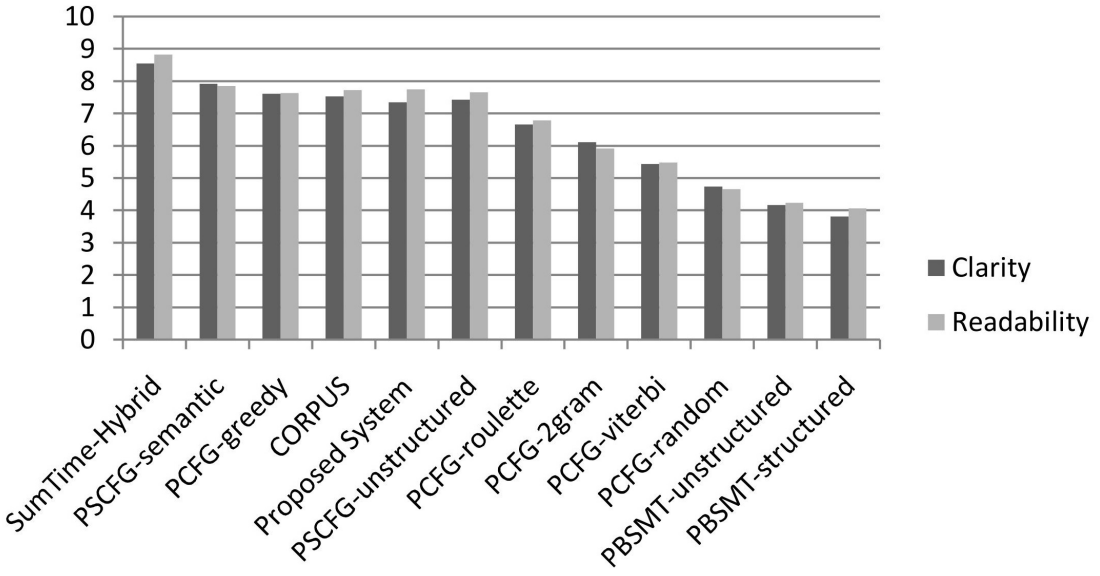

FIGURE 5.3: Comparison of through human-based evaluation

5.3 Result Analysis

As per the outcomes of automatic evaluation, our proposed system provided the third and forth best results in METEOR and BLEU, respectively. According to the METEOR, our system is only behind the PCGF-Greedy and PSCFG-Semantic systems while according to BLEU, PCFG-greedy, PSCFG-Semantic and PSCFG-Unstructured systems perform better than our proposed model. However, these systems which are ahead of our system in performance as per automatic evaluation are not fully automatic, whereas our system does not require any human efforts or external knowledge.

Human evaluation preferred rule based systems over automatic knowledge-light systems. The SumTime system, which is a rule based system and is placed in the ninth position according to both BLEU and METEOR, is adjudged the best system in human evaluation. Our proposed system ranks forth among the 11 systems according to human evaluation. The gold standard reference set was also provided to the human evaluators for evaluation without their knowledge and, surprisingly, the reference set was ranked fourth.

We calculated Pearson correlation coefficient between scores produced by automatic evaluation metrics (BLEU and METEOR) and human evaluation. For human evaluation we considered the average of clarity and readability. The correlation coefficients were r(Human, Bleu)=0.61 and r(Human, METEOR)=0.57 which can be considered as high correlation.

Chapter 6

Conclusions and Future Work

6.1 Conclusions

In this thesis work we have presented a automatic statistical domain independent natural language generation system, which falls in to knowledge light NLG system category. The proposed NLG system doesn't require any sort of human or application-domain expert efforts and helps. Another advantage of our proposed system is this system don't need any changes to shift it from one application domain to another, only a parallel dataset/corpus of new application domain area is required. Whereas others NLG system may need huge modification in that case.

We evaluated our system through both automatic and human evaluation techniques for checking informative and linguistic quality of generated natural language text data. For the confusion of correlation between automatic and human evaluation, we also have computed Pearson correlation term between those two evaluations and we have found that the correlation value is also much greater that -1.

To avail good quality output text from the proposed system, one must conform to the requirement specified in section 3.2. The NLG system will perform accurately if all attributes present in the training tuple-formed non-textual data contain distinct values. If this criterion can be assured, then it will be trivial to match each of those present attribute values to their appearance in the corresponding textual data. However, if this constraint isn't possible to be satisfied our proposed system will still work. It is worth to be mentioned here that, in situations when not a single attribute value of non-textual data

can be found in the corresponding textual data, then our proposed system will behave like an instance based NLG system.

6.2 Future Work

The natural language generation system and its approach presented in this thesis opens up many avenues for future work. Based on the proposed system and its principles discussed in this thesis work, the following future research ideas can be pursued.

- As discussed in Section 3.2, the proposed system is based on the assumption that most of the attribute values present in any non-textual data should also appear in the corresponding textual data. However, this assumption is not true for all datasets or domains. A numeric attribute value might not be directly present in the textual data as it is, instead it might be represented in words. e.g., 00:00am might be represented as 'midnight' in text. Proper alignment of attribute values in the non-textual data to words in the corresponding textual data would improve the results for such cases.
- We mainly used our system for generation of textual data from tuple-formed non-textual data. But, the proposed system can be applied to any dialog system's NLG component, in that case all dialog acts of a dialog system need to be converted into tuple-formed representation.
- As we have already seen in Section 4.2.3 that there are huge number of possible sentences that can be generated for a single non-textual tuple-formed data. There is always scope for improving the search policy for generating better sentences.
- In the work reported in this thesis we carried out our experiments on a wind forecast dataset. In future we would like to carry out experiments on some other datasets from other domains.

Bibliography

[1] Eli Goldberg, Norbert Driedger, and Richard I. Kittredge. Using natural-language processing to produce weather forecasts. *IEEE Expert: Intelligent Systems and Their Applications*, 9(2):45–53, April 1994. ISSN 0885-9000. doi: 10.1109/64.294135. URL http://dx.doi.org/10.1109/64.294135.

[2] Bruce G. Buchanan, Johanna D. Moore, Diana E. Forsythe, Giuseppe Carenini, Stellan Ohlsson, and Gordon Banks. An intelligent interactive system for delivering individualized information to patients. *Artificial Intelligence in Medicine*, 7(2):117 – 154, 1995. ISSN 0933-3657. doi: http://dx.doi.org/10.1016/0933-3657(94)00029-R. URL http://www.sciencedirect.com/science/article/pii/093336579400029R.

[3] W. R. Swartout. XPLAIN: A system for creating and explaining expert consulting programs. *Artificial Intelligence*, 21(3):285–325, September 1983.

[4] Ehud Reiter and Robert Dale. Building applied natural language generation systems. *Nat. Lang. Eng.*, 3(1):57–87, March 1997. ISSN 1351-3249. doi: 10.1017/S1351324997001502. URL http://dx.doi.org/10.1017/S1351324997001502.

[5] Ibrahim Adeyanju. Article: Generating weather forecast texts with case based reasoning. *International Journal of Computer Applications*, 45(10):35–40, May 2012. Full text available.

[6] Ehud Reiter, Somayajulu Sripada, Jim Hunter, Jin Yu, and Ian Davy. Choosing words in computer-generated weather forecasts. *Artif. Intell.*, 167(1-2):137–169, September 2005. ISSN 0004-3702. doi: 10.1016/j.artint.2005.06.006. URL http://dx.doi.org/10.1016/j.artint.2005.06.006.

[7] Robert Dale and Ehud Reiter. Computational interpretations of the gricean maxims in the generation of referring expressions. *CoRR*, cmp-lg/9504020, 1995. URL http://arxiv.org/abs/cmp-lg/9504020.

[8] Stephan Busemann. Best-first surface realization. In *Proceedings of the 8th. International Workshop on Natural Language Generation (INLG '96)*, pages 101–110, Herstmonceux, England, June 1996. URL http://xxx.lanl.gov/abs/cmp-lg/9605010.

[9] Benoit Lavoie and Owen Rambow. A fast and portable realizer for text generation systems. In *Proceedings of the Fifth Conference on Applied Natural Language Processing*, ANLC '97, pages 265–268, Stroudsburg, PA, USA, 1997. Association for Computational Linguistics. doi: 10.3115/974557.974596. URL http://dx.doi.org/10.3115/974557.974596.

[10] Michael Elhadad and Jacques Robin. An overview of surge: a reusable comprehensive syntactic realization component. Technical report, 1996.

[11] M. A. K. Halliday and Christian M. I. M. Matthiessen. *An introduction to functional grammar / M.A.K. Halliday*. Hodder Arnold London, 3rd ed. / rev. by christian m.i.m. matthiessen. edition, 2004. ISBN 0340761679 9780340761670.

[12] Ehud Reiter. An architecture for data-to-text systems. In *Proceedings of the Eleventh European Workshop on Natural Language Generation*, ENLG '07, pages 97–104, Stroudsburg, PA, USA, 2007. Association for Computational Linguistics. URL http://dl.acm.org/citation.cfm?id=1610163.1610180.

[13] Anja Belz. Automatic generation of weather forecast texts using comprehensive probabilistic generation-space models. *Nat. Lang. Eng.*, 14(4):431–455, October 2008. ISSN 1351-3249. doi: 10.1017/S1351324907004664. URL http://dx.doi.org/10.1017/S1351324907004664.

[14] Kathleen McKeown, Karen Kukich, and James Shaw. Practical issues in automatic documentation generation. In *Proceedings of the Fourth Conference on Applied Natural Language Processing*, ANLC '94, pages 7–14, Stroudsburg, PA, USA, 1994. Association for Computational Linguistics. doi: 10.3115/974358.974361. URL http://dx.doi.org/10.3115/974358.974361.

[15] Anja Belz and Eric Kow. System building cost vs. output quality in data-to-text generation. In *Proceedings of the 12th European Workshop on Natural Language*

Generation, ENLG '09, pages 16–24, Stroudsburg, PA, USA, 2009. Association for Computational Linguistics. URL http://dl.acm.org/citation.cfm?id=1610195.1610198.

[16] Yuk Wah Wong and Raymond J. Mooney. Generation by inverting a semantic parser that uses statistical machine translation. In *Proceedings of Human Language Technologies: The Conference of the North American Chapter of the Association for Computational Linguistics (NAACL-HLT-07)*, pages 172–179, Rochester, NY, 2007. URL http://www.cs.utexas.edu/users/ai-lab/?wong:naacl07.

[17] Kevin Knight and Vasileios Hatzivassiloglou. Two-level, many-paths generation. In *Proceedings of the 33rd Annual Meeting on Association for Computational Linguistics*, ACL '95, pages 252–260, Stroudsburg, PA, USA, 1995. Association for Computational Linguistics. doi: 10.3115/981658.981692. URL http://dx.doi.org/10.3115/981658.981692.

[18] 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*, pages 704–710, Montreal, Quebec, Canada, August 1998. Association for Computational Linguistics. doi: 10.3115/980845.980963. URL http://www.aclweb.org/anthology/P98-1116.

[19] Nizar Habash. *Envisioning Machine Translation in the Information Future: 4th Conference of the Association for Machine Translation in the Americas, AMTA 2000 Cuernavaca, Mexico, October 10–14, 2000 Proceedings*, chapter Oxygen: A Language Independent Linearization Engine, pages 68–79. Springer Berlin Heidelberg, Berlin, Heidelberg, 2000. ISBN 978-3-540-39965-0. doi: 10.1007/3-540-39965-8_7. URL http://dx.doi.org/10.1007/3-540-39965-8_7.

[20] Brian Langner and Alan W Black. Mountain: a translation-based approach to natural language generation for dialog systems. 2009.

[21] Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondřej Bojar, Alexandra Constantin, and Evan Herbst. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th*

Annual Meeting of the ACL on Interactive Poster and Demonstration Sessions, ACL '07, pages 177–180, Stroudsburg, PA, USA, 2007. Association for Computational Linguistics. URL http://dl.acm.org/citation.cfm?id=1557769.1557821.

[22] Shimei Pan and James Shaw. Segue: A hybrid case-based surface natural language generator. In *In Proceedings of INLG 2004*, pages 130–140, 2004.

[23] Tomas Mikolov, Martin Karafiát, Lukás Burget, Jan Cernocký, and Sanjeev Khudanpur. Recurrent neural network based language model. In *INTERSPEECH 2010, 11th Annual Conference of the International Speech Communication Association, Makuhari, Chiba, Japan, September 26-30, 2010*, pages 1045–1048, 2010. URL http://www.isca-speech.org/archive/interspeech_2010/i10_1045.html.

[24] Ilya Sutskever, James Martens, and Geoffrey Hinton. Generating text with recurrent neural networks.

[25] Tsung-Hsien Wen, Milica Gasic, Dongho Kim, Nikola Mrksic, Pei-hao Su, David Vandyke, and Steve J. Young. Stochastic language generation in dialogue using recurrent neural networks with convolutional sentence reranking. *CoRR*, abs/1508.01755, 2015. URL http://arxiv.org/abs/1508.01755.

[26] F. Jelinek. Fast sequential decoding algorithm using a stack. *IBM J. Res. Dev.*, 13 (6):675–685, November 1969. ISSN 0018-8646. doi: 10.1147/rd.136.0675. URL http://dx.doi.org/10.1147/rd.136.0675.

[27] Anja Belz. Prodigy-meteo: Pre-alpha release notes (nov 2009). 2009.

[28] 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 on Association for Computational Linguistics*, ACL '02, pages 311–318, Stroudsburg, PA, USA, 2002. Association for Computational Linguistics. doi: 10.3115/1073083.1073135. URL http://dx.doi.org/10.3115/1073083.1073135.

[29] Satanjeev Banerjee and Alon Lavie. Meteor: An automatic metric for mt evaluation with improved correlation with human judgments. pages 65–72, 2005.