Pilot Operating Handbook Semantic Text Analysis

Project Requirements and Specifications

**The Boeing Company**

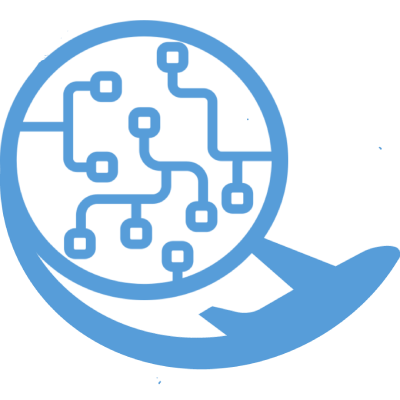


**The Autopilots**

Katie Peterson

Allison Lorphanpaibul

Weiren Lai

Daniel Harker

October 6, 2020

**TABLE OF CONTENTS**

[**Introduction**](#_heading=h.e0ks7qwbdnp0) **3**

[**System Requirements Specification**](#_heading=h.3znysh7) **3**

[Use Cases](#_heading=h.2et92p0) 3

[Functional Requirements](#_heading=h.tyjcwt) 5

[*System Input & Output Requirements*](#_heading=h.3dy6vkm) *5*

[*System Parsing & Identification Requirements*](#_heading=h.1t3h5sf) *5*

[Non-Functional Requirements](#_heading=h.4d34og8) 5

[**System Evolution**](#_heading=h.2s8eyo1) **6**

[**Glossary**](#_heading=h.17dp8vu) **7**

[**References**](#_heading=h.3rdcrjn) **7**

# Introduction

Artificial intelligence (AI) and machine learning (ML) have become the trends of the future; however, the cost of AI and ML is very high, especially in acquiring and maintaining real and production data. Hence, there is a need to be able to support training synthetic data engines like GANs (generative adversarial networks) to create synthetic data on-demand. These engines provide a fast and inexpensive way to produce data that is properly marked and tagged, unlike most real and production data.

Our goal is to develop a semantic text parsing algorithm that takes vocabulary from Boeing aircraft handbooks and stores each noun/noun phrase with its designated document name, year, product, location, and 1-50 sentences to demonstrate context for each noun. We expect the program to have a 60-90% quality after training the GAN [1].

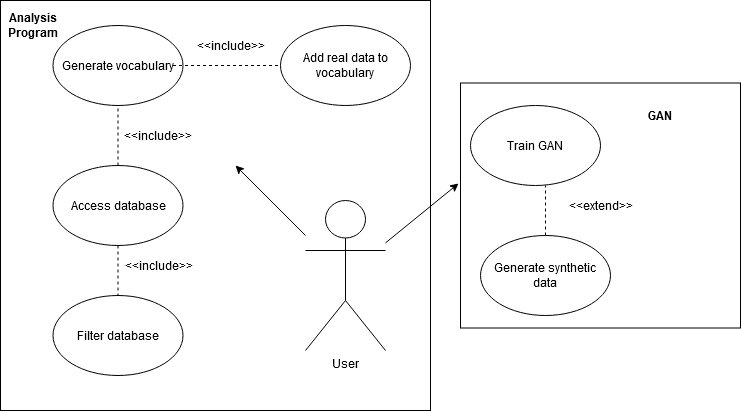
# System Requirements Specification

## Use Cases

|  |  |
| --- | --- |
| Use Case | User generates a constrained vocabulary |
| Pre-condition | (none) |
| Actions | 1. User inputs document written in simplified english 2. System responds by parsing the document 3. System puts parsed vocabulary into database/spreadsheet 4. (optional) User inputs vocabulary as training data for GAN |
| Post-condition | Database has new entry with parsed vocabulary |
| Acceptance tests | Database entry exists |
| Requirements | * Algorithm for noun parsing * Identification of noun information |

|  |  |
| --- | --- |
| Use Case | User filters constrained vocabulary lists |
| Pre-condition | Database/spreadsheet with vocabularies has been generated from the program. |
| Actions | 1. User opens up database/spreadsheet. 0 2. User can query from data using different filters, such as year, date, document, etc using options provided. 3. The system responds with showing the filtered lists |
| Post-condition | User can easily filter different vocabulary lists |
| Acceptance tests | Correct data is shown |
| Requirements | * Data is stored in a database/spreadsheet * Identification of noun information |

|  |  |
| --- | --- |
| Use Case | User adds real/production data to vocabulary |
| Pre-condition | Database/spreadsheet with vocabularies has been generated from the program. |
| Actions | 1. User inputs vocabulary and real data 2. System produces synthetic data 3. User inputs data for GAN training |
| Post-condition | Vocabulary will deliver 90% quality test data when training the GAN. |
| Acceptance tests | New test data delivers 90% quality |
| Requirements | * Data is stored in a database/spreadsheet |

* + 1. **Use Case Diagram**

Above shows the two different areas of use cases for a user.

Our text analysis program (left) will work independently from the GAN (right).  
The GAN program and real data will be provided by Boeing.

## Functional Requirements

### System Input & Output Requirements

**System Input:** The system must read as input 3-10 Boeing pilot operating handbooks in PDF format.

**Source**: Rakshit Bhatt and Don Brancato of The Boeing Company. Necessary to ensure the nouns that are parsed are used in the correct context for Boeing.

**Priority**: Level 0 - Essential and required functionality

**System Output:** The system must store data collected in a MySQL database or Open Office Spreadsheet.

**Source**: Rakshit Bhatt and Don Brancato of The Boeing Company. Necessary to ensure the data can be located and used to train GANs in the future.

**Priority**: Level 0 - Essential and required functionality

### System Parsing & Identification Requirements

**Algorithm for Noun Parsing:** The system must have a semantic text parsing algorithm, capable of parsing all nouns from text written in Boeing Simplified English.

**Source**: Rakshit Bhatt and Don Brancato of The Boeing Company. Necessary to ensure nouns are collected for use in training GANs.

**Priority**: Level 0 - Essential and required functionality

**Identification of Noun Information:** For each unique noun parsed, the system must also identify the following: 1 to 50 sentences from the various documents that contain the noun, the name of the document(s) the noun appears in, the product associated with the document(s), the publication year of the document(s), and the location of the document(s).

**Source**: Rakshit Bhatt and Don Brancato of The Boeing Company. Necessary to ensure the GANs are trained with context for each noun.

**Priority**: Level 0 - Essential and required functionality

## Non-Functional Requirements

**Reliability/Performance – Accuracy Percentages for Real and Synthetic Data**

The program must be able to consistently produce accurate results for the given data. When testing the GANs with real data, we expect to have 40-70% accuracy. When pairing real data with synthetic, this accuracy should jump up to 90%.

**Response Time – Quickly Generate Constrained Vocabulary**

The program needs to be able to generate a constrained vocabulary quickly. Since there are thousands of simplified languages at Boeing, this program will need to be able to create the vocabulary in a timely manner so that it can be used efficiently.

**Readability – Create Simple Databases of Nouns**

With the input to the program being Boeing-provided documents and manuals that are hundreds of pages long, the output needs to be clear and concise. This output will take the form of a database of nouns that can be read by humans and parsed by computers.

**Usability – Straight-forward Creation of Simplified Languages by User**

The program interface needs to be designed so that any user can select multiple files that will be turned into a simplified language. This design needs to be uncomplicated and straightforward to allow for ease of use.

# System Evolution

**Continuing change**

We are aware of how systems and tools are evolving in the future, continuing change and updating will be one of our priorities in maintaining the quality of the program.

**Increasing complexity**

Mode collapse issues will become problematic in GANs training later on. The learning process of GANs might have a missing pattern, the generator begins to degenerate, the learning will not be continued because the same sample points are getting generated.[3]

However, we could force the generator to broaden its scope by preventing it from optimizing for a single fixed discriminator. Wasserstein loss slows down mode collapse by letting us train the discriminator to optimality without worrying about vanishing gradients. It means if the discriminator doesn’t get stuck in local minima, it learns to reject outputs that the generator stabilizes on, hence the generator will have to try something new. Another way is unrolled GANs. It uses a generator loss function that takes in not only the current discriminator’s classification, but also the outputs of future discriminator versions. Therefore, the generator will not be able to over-optimize for a single discriminator.[2]

**Text parsing algorithm**

When it comes to parsing, ambiguity in parsing becomes a problem; A sentence is structurally ambiguous if the grammar assigns it more than a possible parse. There are two kinds of parsing strategies in terms of parsing: top-down, bottom-up. Top down strategy may have problems such as left recursion, left factoring, backtracking and ambiguity. Bottom-up strategy may develop issues such as sometimes parsers can reduce but it should not, and it could reduce in different ways, and how do we know whether to shift or reduce, which production to use for reduction.

**Database**

We need to decide if we want to use a spreadsheet or a database. If we use a database, when the data gets larger we will have to decide whether to use a cloud database or a local database.

**Platform**

Now we are planning to use programming languages like Python, Scala, but later on when we dig deeper, we might have new ideas of which programming language to use in terms of complexity, memory use etc.. We will be ready to shift our work when it’s needed.

# Glossary

**Generative Adversarial Networks (GANs) -**  A popular machine learning model used to create synthetic data on-demand.

**Simplified English -** The standard for written aerospace documents. Utilizes a restricted vocabulary and simple grammar rules.

# References

[1] Rakshit Bhatt & Don Brancato, “Pilot Operating Handbook Semantic Text Analysis”, The Boeing Company, Washington, USA, 2020.

[2] “Common Problems | Generative Adversarial Networks | Google Developers.” *Google*, Google, developers.google.com/machine-learning/gan/problems.

[3] die, Xiaoqiang who can't. “Understand the Basic Principles of ‘Generation Adversarial Network-GAN’ + 10 Typical Algorithms + 13 Applications.” *产品经理的人工智能学习库*, 17 Dec. 2019, easyai.tech/en/ai-definition/gan/.