TU WIEN Informatics

Graphenbasierte Methoden zur Klassifizierung von Nutzerabsichten

DIPLOMARBEIT

zur Erlangung des akademischen Grades

Diplom-Ingenieur

im Rahmen des Studiums

Data Science

eingereicht von

Mal Kurteshi, B.Sc
Matrikelnummer 11924480

an der Fakultät für Informatik

der Technischen Universität Wien

Betreuung: Univ.Prof.Dr. Allan Hanbury
Mitwirkung: Univ.Ass.PhD Gábor Recski

Wien, 16. März 2023

Mal Kurteshi Allan Hanbury

TU WIEN Informatics

Graph-based methods for user intent classification

DIPLOMA THESIS

submitted in partial fulfillment of the requirements for the degree of

Diplom-Ingenieur

in

Data Science

by

Mal Kurteshi, B.Sc
Registration Number 11924480

to the Faculty of Informatics

at the TU Wien

Advisor: Univ.Prof.Dr. Allan Hanbury
Assistance: Univ.Ass.PhD Gábor Recski

Vienna, 16th March, 2023

	Mal Kurteshi	Allan Hanbury

Erklärung zur Verfassung der Arbeit

Mal Kurteshi, B.Sc

Hiermit erkläre ich, dass ich diese Arbeit selbständig verfasst habe, dass ich die verwendeten Quellen und Hilfsmittel vollständig angegeben habe und dass ich die Stellen der Arbeit – einschließlich Tabellen, Karten und Abbildungen –, die anderen Werken oder dem Internet im Wortlaut oder dem Sinn nach entnommen sind, auf jeden Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht habe.

Wien, 16. März 2023

Mal Kurteshi

Danksagung

Zunächst möchte ich mich bei der Technischen Universität Wien für die Möglichkeit und die Ehre bedanken, als Masterstudent Teil dieser großartigen Universität zu sein.
Mein aufrichtiger Dank gilt meinem Betreuer Univ.Prof.Dr. Allan Hanbury und Co-Betreuer Univ.Ass.PhD Gábor Recski. Diese Arbeit wäre ohne die Hilfe, Unterstützung und Ermutigung von Professor Recski während des Prozesses nicht durchführbar gewesen. Ich werde ihm immer dankbar sein.
Ich möchte meiner Familie für ihre bedingungslose Unterstützung während dieser Zeit danken. Obwohl sie weit weg von Wien waren. Sie haben nie aufgehört, mich zu lieben und mich während meines Studiums zu unterstützen. Vielen Dank dafür!
Zu guter Letzt möchte ich meiner Verlobten Yllka danken, nicht nur für ihre bedingungslose Liebe, sondern auch für ihre Unterstützung, Ermutigung und dafür, dass sie immer an mich geglaubt hat. Danke, meine Liebe!

Acknowledgements

First, I would like to thank the Technical University of Vienna for the opportunity and honor of being part of this great university as a master's student.

My sincere appreciation goes to my supervisor Univ.Prof.Dr. Allan Hanbury and co-supervisor Univ.Ass.PhD Gábor Recski. This thesis would not have been feasible without professor Recski's assistance, support, and encouragement. I will always be grateful.

I want to thank my family for their unconditional support during these times. Although they were far away from Vienna. They never stopped loving and supporting me during my studies. Thank you!

Last but not least, I want to thank my fiance Yllka, not only for her unconditional love but also for her support, encouragement, and for always believing in me. Thank you, my love!

Kurzfassung

In dieser Masterarbeit stellen wir ein hybrides System zur Absichtsklassifizierung vor, das auf einer auf einer syntaktischen Graphendarstellung natürlicher Sprache basiert.
Unser Ziel war es, ein System zu entwickeln, das die Stärken von regelbasierten Ansätzen mit Modellen des maschinellen Lernens, insbesondere Support Vector Machines (SVM) und Bidirectional Encoder-Repräsentationen von Transformatoren (BERT).
Unser Ziel war es, ein System zu entwickeln, das Benutzerabsichten in natürlicher Sprache genau klassifizieren kann. Sprache klassifiziert.

Um die Wirksamkeit des Systems zu bewerten, haben wir sowohl qualitative als auch quantitative Methoden.
Die qualitative Analyse konzentrierte sich auf die Darstellung des syntaktischen Graphen und seine Fähigkeit, komplexe komplexe Sprachstrukturen zu erfassen.
Wir fanden heraus, dass die syntaktische Graphendarstellung die semantische Bedeutung des Bedeutung des Eingabetextes zu erfassen, was eine genaue Klassifizierung der Absicht ermöglichte.
Das regelbasierte System, das auf der Darstellung des syntaktischen Graphen basierte bei einigen Absichten gut, bei anderen jedoch weniger gut.

Deshalb haben wir ein Hybridsystem entwickelt, das den regelbasierten Ansatz mit maschinellen Lernmodellen kombiniert.
Die quantitative Analyse ergab, dass das SVM-Modell versteckte Verzerrungen gegenüber bestimmten Absichten hatte, was seine Gesamtleistung beeinträchtigte.
Andererseits schnitt das BERT-Modell besser ab als das SVM-Modell, mit einem leichten Unterschied zum Hybridmodell.
Das Hybridsystem war in der Lage, die Stärken des regelbasierten Ansatzes und der Modelle des maschinellen Lernens zu kombinieren, was zu einer verbesserten Leistung in allen Intents führte.

Unsere Ergebnisse unterstreichen die Bedeutung der qualitativen Analyse bei der Entwicklung effektiver Systeme zur Verarbeitung natürlicher Sprache.
Wenn wir die syntaktische Struktur der natürlichen Sprache verstehen, können wir bessere Modelle erstellen die die Bedeutung des Eingabetextes genau erfassen.

Darüber hinaus zeigt das von uns entwickelte Hybridsystem, dass es die Genauigkeit und Robustheit von Systemen zur Klassifizierung von Absichten verbessern kann.

Zusammenfassend lässt sich sagen, dass unsere Arbeit einen Einblick in die Effektivität eines hybriden Systems zur das die Stärken von regelbasierten und maschinellen Lernansätzen kombiniert.Ansätze kombiniert.
Die Ergebnisse dieser Studie haben praktische Auswirkungen auf die Entwicklung von genaueren und robusteren und robusten Systemen zur Absichtsklassifikation, die die Leistung verschiedener Anwendungen für natürliche Sprache verbessern können.
Unsere Arbeit trägt zu den laufenden Bemühungen bei, Systeme zur Verarbeitung natürlicher Sprache zu entwickeln Systeme zu entwickeln, die die Absichtsklassifikation genau und effektiv verarbeiten können.

Abstract

In this master's thesis, we present a Hybrid system for intent classification based on a syntactic graph representation of natural language.
We aimed to create a system that combined the strengths of rule-based approaches with machine learning models, specifically Support Vector Machines (SVM) and Bidirectional Bidirectional Encoder Representations from Transformers (BERT).
Our goal was to develop a system that could accurately classify user intents in natural language.

We used qualitative and quantitative methods to evaluate the system's effectiveness.
The qualitative analysis focused on syntactic graph representation and its ability to capture complex language structures.
We found that the syntactic graph representation effectively captured the semantic meaning of the input text, enabling accurate intent classification.
However, based on the syntactic graph representation, the rule-based system performed well on some intents but was less effective on others.

Therefore, we developed a Hybrid system that combined the rule-based approach with machine learning models.
The quantitative analysis revealed that the SVM model had hidden biases towards certain intents, which affected its overall performance.
On the other hand, the BERT model performed better than the SVM model with a slight difference from the Hybrid model.
The Hybrid system combined the strengths of the rule-based approach and machine learning models, resulting in improved performance across all intents.

Our findings highlight the importance of qualitative analysis in developing effective natural language processing systems.
By understanding the syntactic structure of natural language, we can create better models that accurately capture the meaning of the input text.
Moreover, the Hybrid system we developed shows promise in improving the accuracy and robustness of intent classification systems.

In conclusion, our thesis provides insights into the effectiveness of a Hybrid system for intent classification that combines the strengths of rule-based and machine-learning approaches.

The results of this master thesis have practical implications for developing more accurate and robust intent classification systems, which can improve the performance of various natural language applications.

Our work contributes to the ongoing efforts to develop natural language processing systems that accurately and effectively process intent classification.

Contents

CHAPTER 1

Introduction

In this master's thesis, we propose an approach to user intent classification using syntactic graphs with linguistic meaning representations and graph rule-based methods.

User intent is the recognition and classification of what a user meant or wanted to discover when they delivered their sentence, speech, or state into an environment.

Let us take a small example of user intent. Let us assume that an airplane agency had a phone call, and from the transcript, the user said, *"What is the arrival time in San Francisco for the 7:55 am flight leaving Washington? [MS CNTK19]"* If we analyze this sentence, we can assume that the user intent is the *"flight time"* of the particular flight. From this assumption, we classify user intent from the user sentence into one category, "flight time."

The above example and examples we present in this master's thesis report are from the Airline Travel Information Systems (ATIS)[MS CNTK19] dataset.

The ATIS[MS CNTK19] dataset includes audio recordings of people requesting flight information and the corresponding manual transcripts [SYG19].

Intent classification matches words or sentences with a particular intent through machine learning and natural language processing. We will use graph-based methods and compare them with machine learning models such as SVM and BERT.

The advantages of graph-based methods are the ability to change the model easily if an error occurs. Also, graph-based methods reveal visible and explicit biases, as machine learning approaches can have hidden biases.

The disadvantages of the graph-based models are that they are hard to maintain and have worse performance on the benchmark.

While Machine Learning (ML) and Deep Learning (DL) outperform benchmarks, they also have the disadvantage of producing hidden biased results.

Therefore, for user intent classification, our comparison of graph-based methods and ML or DL takes the basis of their advantages and disadvantages.

1.1 Aims of this Thesis

This thesis aims to develop a rule-based solution to model semantic graph tasks with good precision comparable to Machine Learning.
This approach is a way of building a rule-based system that uses semantic representation, and for a study case, we have picked the user intent classification task.
To achieve this, we first need an initial understanding of the representation and parsing of the data on syntactic graphs. Then we define rules from graphs to build our rule-based system. Furthermore, we will establish a baseline to compare the proposed rule-based approach with machine learning techniques like SVM and BERT.
For this purpose, we are using the dataset ATIS [MS CNTK19]. Furthermore, we will investigate the dataset's structure and understand the data distribution and insights that can be obtained from the dataset.
We will use syntactic graphs to represent the data.
Syntactic graph representation makes it possible to represent all possible surface syntactic relations in one directed graph [SS89].
Furthermore, from the graph-based system, we expect advantages such as straightforward interpretation and explanation by user design, less Graphical Processing Unit (GPU) resource for training, a fully customized model, and an easy debugging model from the user [DVK17].
Although on the other hand, disadvantages like difficulty in maintenance and the need for more expertise to develop them compared to Machine learning approaches [DVK17].
We anticipate that the outcomes of the machine learning technique could suffer from biased results [DVK17].
For instance, if we have set lots of samples in training with the sentence *"boeing777 landed on 20:00"* [MS CNTK19]. These samples are related to the type of aircraft which landed. Then the model can be biased to classify all aircraft that land at 20:00 as *boeing777*, which is not necessarily true and presents a hidden model bias that can be treated with rule-based systems.
To conclude, we will answer the following questions.

1. How does a rule-based system using syntactic graphs perform on the intent classification task?

2. How do graph-based methods compare to simple ML baselines?

3. What are the bottlenecks of rule-based systems, and what syntactic patterns characterize the main error classes?

1.2 Contribution

The main contributions of this thesis are:

- An in-depth, comprehensive review covering both practical and theoretical aspects of the latest rule-based system frameworks, highlighting their state-of-the-art features and advancements.

- Using Universal Dependencies (UD) graphs to understand the syntactic relations on a sentence and then creating rules based on these graphs, we have developed a rule-based system for the intent classification task. Our approach leverages the hierarchical structure of the dependency tree to generate rules from raw data.

- Performance evaluation of our system is done using various metrics and analyzing the different components to assess their effectiveness in intent classification.

- Under controlled experimental conditions, our rule-based system achieved results comparable or superior to those obtained by state-of-the-art frameworks for most of the dataset examined.

- The hybrid system we developed, which combines SVM and graph-based rule methods, produced lower error rates than when the systems were used separately. Our evaluation of the system's performance on ATIS dataset demonstrated its potential for achieving superior results in the intent classification tasks.
  Furthermore, our analysis of the system's components and metrics revealed that combining SVM and graph rules offered complimentary benefits contributing to its improved performance.

1.3 Organization

The second chapter provides an overview of the user intent classification concept, which involves identifying the intention behind a user's input or request.
To contextualize this notion, we review existing literature on similar studies and present in-depth research on user intent classification. By examining the strengths and limitations of this approach, we aim to provide a comprehensive understanding of the state-of-the-art techniques in the field and identify potential opportunities for further research and development.
The third chapter discusses the use of graphs for representing sentences and explains why graphs are beneficial for this task.
The advantages of using UD graphs for intent classification. The concept of a rule-based approach to intent classification and its advantages and disadvantages are introduced.
This chapter describes how rules can be parsed on graphs and how graphs analysis can help identify new patterns that can be mapped to rules. Additionally, the section covers the use of Explainable Artificial Intelligence (XAI) frameworks like POTATO[KGIR22] to apply machine learning to rule predictions based on intent features and the importance of expertise in the field to define rules.
The fourth chapter will focus on the Machine Learning and Deep Learning approach to Intent classification, specifically the SVM and BERT algorithms.

The chapter will delve into the similarities and differences between these two algorithms, highlighting their strengths and weaknesses. An overview of the SVM algorithm will be presented, including how the model is defined and trained. In contrast, the BERT model, a neural network approach to intent classification, will also be described.
The chapter will also discuss the stopping criteria for determining when the model is not overfitting or underfitting, as well as the black box effect on the hidden layers of the model.
In the fifth chapter, we present the results obtained.
The results present quantitative and qualitative analyses. First, we provide a detailed comparison of the performance of our system with state-of-the-art machine learning models such as SVM and BERT.
We propose a hybrid model between SVM and a rule-based system. Finally, we provide qualitative insights into the strengths and limitations of our proposed system, including its interpretability, scalability, and generalizability.
In chapter six, concluding remarks are given.

CHAPTER 2

Problem Statement and Related Work

2.1 Problem Statement

Intent classification is a significant task in spoken language knowledge, and part of Natural Language Processing (NLP), which focuses on classifying text for a better understanding of the text's meaning [SHRJ21].
For example, the sentence *"What flights are available from Pittsburgh to Baltimore on Thursday morning? [MS CNTK19]"* indicates a flight request and can be classified as a *"flight request" [MS CNTK19].*
ML models currently dominate text processing tasks. Several approaches with machine learning have been applied for user intent classification and show promising results on the benchmark. However, as the parameters of these models increase exponentially, their explainability decreases.
Another critical problem with the machine learning approach is the possibility of producing hidden biased results.
So with lower explainability and potential hidden biases, the machine learning approach to text processing tasks also gives the possibility of suffering from generalization, which presents the model's ability to adapt to new unseen data.
The graph-based approach is suitable for fixing the above premises. Moreover, the graph-based approach allows us to define rules from graphs, which will resolve the problem of low explainability, provide less biased predictions and minimize the generalization problems of predictions on unseen data.

2.2 Related Work

There are already some studies conducted on user intent classification with exciting results and findings, which helped us identify state-of-the-art research and formulate the problem definition.
Chen et al. [CZW19] propose a joint intent classification and slot-filling model based on BERT, aiming at addressing the poor generalization capability of traditional Natural Language Understanding (NLU) models [CZW19]. The experiment's results show the efficacy of exploiting the relationship between intent classification and slot filling. Furthermore, accuracy and F1 score show excellent results over other model comparisons. In future work, we can test the model's performance on larger scales where we could face biased results.
Mehrabi et al. [MMS$^+$]introduce problems that can affect machine learning and natural language processing regarding unfairness and bias. They present different sources and types of biased predictions on machine learning systems. Regarding classification fairness, since classification is a canonical task in machine learning and is widely used in different areas that can be in direct contact with humans, these methods must be fair and absent of biases that can harm some populations [MMS$^+$].
A general methodology for dealing with bias in Deep NLP is presented by Garrido-Muñoz et al. [GMMRMSUL]. This methodology consists of modifying the training corpora, the training algorithm, or the results obtained according to the given task [GMMRMSUL]. Garrido-Muñoz et al. [GMMRMSUL] proposition is to systematize the evaluation of the impact of bias as part of the design of systems relying on deep NLP techniques and resources.
The issue with biased prediction and unfairness also affect the model's interpretability. For example, Schnack et al. [Sch] show that in these terms, the selection of features and the selection of the machine learning algorithm will affect the interpretability of the model.
Since the machine learning approach can have hidden biases leading to problems with fairness and model interpretability, the rule-based approach helps us avoid these cases. For rule-based systems, there is an explainable information extraction framework named POTATO with which we can determine rule-based systems.
Kovacs et al. [KGIR22] present the usage of the POTATO framework, its flexibility in creating rule-based and graph rule-based models, difficulties, advantages, and disadvantages of their results.
SVM, among other machine learning methods, address user intent classification problems. Mendoza et al. [MZ09] show the SVM machine learning model approach in identifying the intent of a user query. The SVM method gives good results in the test sample of the dataset but can face difficulties with large-scale data.

CHAPTER 3

Intent classification through Graph-Based methods

This chapter presents our graph-based approach to intent classification, from which we have defined a rule-based model. We have presented how to define rules from graphs and model a rule-based classifier.
Graph-based methods aim to present text as a graph, allowing for the identification of its most effective features and characteristics [OB].
Sentence representation through graphs is a crucial step in our data preprocessing.
Before proceeding with a graph-based method for user intent classification, we need to be able to represent the data in graph format.
But what is a graph or a directed graph?
Let us present these two concepts with the definitions below:

Definition 1 *"A graph is a collection of connections between objects, where the objects are called vertices or nodes, and the connections between them are called edges" [ZBY07].*

Definition 2 *“Let V be a set of vertices and A a set of ordered pairs of vertices, called arcs or directed edges. Then, a directed graph or digraph, short for directed graph, G is an ordered pair G:= (V, A) where V is the set that contains all the vertices that form G, and A is the set that contains all the arcs that form G" [ZBY07].*

In our graph representation of sentences, each word tag is set as a node or vertex.
The connections between these nodes represent the syntactic relationship between them.
Using graphs to represent sentences, we will purchase powerful graph-based algorithms and techniques to extract meaningful patterns and insights from the data, from which we will perform rules-based user intent classification.

3.1 Syntactic graphs

In our sentence analysis, our focus is on the syntactic representation of the sentence. Sentences can be very similar but with different intents. For example, from the ATIS dataset:

- "Show me the flights available from San Francisco to Pittsburgh for Tuesday and also the price [MS CNTK19]." has flight intent.
- "What are the schedule of flights from Boston to San Francisco for august first [MS CNTK19]?" has flight time intent.
- "What are the flight numbers of the flights which go from San Francisco to Washington via Indianapolis [MS CNTK19]?" has flight number intent.

Although all three sentences of this example are similar in meaning, all three represent a request for information.
This way, to create general rules to predict specific intents, using the text's syntactic representation is more accurate instead of focusing just on the sentence's meaning.
The syntactic graph is a dependency structure that represents a text or sentence as a graph, where each word or token in the sentence is a node or vertex in the graph, and the edges between the nodes represent syntactic relationships [OB].
For syntactic graph text representation, we are using UD graphs.
The Universal Dependencies (UD) framework provides a uniform approach for annotating grammar, including parts of speech, morphological features, and syntactic dependencies, in various human languages with consistency [niv].
UD syntactic graph text representation has several advantages and disadvantages.
Advantages:

- Provides a consistent and unified annotation system for part of speech, morphological features, and syntactic dependencies, making comparing and analyzing text data easier.
- Allows for easy visualization and analysis of sentence structures and dependencies.
- Provides a standard format that can be easily used in various NLP applications.

Disadvantages:

- The annotation process can be time-consuming and resource-intensive, especially for languages with complex syntax.
- It does not capture part of the semantic meaning of a sentence, as it focuses on syntactic relationships between words rather than their semantic relationships.

To provide a graphical representation of UD we are using ***networkx***[HSS08] python package but also ***stanza*** [QZZ$^+$20].
Stanza, an NLP analysis package in python, can identify named entities and produce a syntactic structure dependency parse [QZZ$^+$20].
It provides tools that transform human language text sequentially into sentences and words and generate their base forms, parts of speech, and morphological features [QZZ$^+$20].
NetworkX is a Python language package for the exploration and analysis of networks and network algorithms[HSS08].
Let us take an example of UD syntactic graph text representation.
We have the text "show me the flights available from San Francisco to Pittsburgh for Tuesday and also the price[MS CNTK19]" then the syntactic representation of this text using UD graphs will be like below:

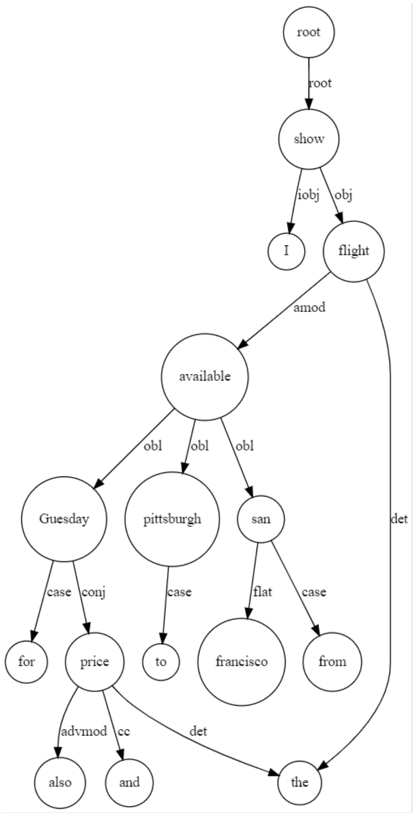


Figure 3.1: Syntactic graph text representation

UD graph representation of a sentence forms a tree, where precisely one word is the head of the sentence, marked as "root," and all other words depend on another. In our case, the head of the sentence is the word tag "show".
The table 3.1 shows the syntactic dependency between two-word tags in our example. We are using these syntactic dependencies between words in our approach to defining

Dependency type	Description [niv]
iobl - oblique nominal	Used for a nominal (noun, pronoun, noun phrase) functioning as a non-core (oblique) argument or adjunct
obj - object	The object of a verb. It is the noun phrase that denotes the entity acted upon or which undergoes a change of state or motion
amod - adjectival modifier	Is any adjectival phrase that serves to modify the noun (or pronoun)
det - determiner	Holds between a nominal head and its determiner
obl - oblique nominal	Used for a nominal (noun, pronoun, noun phrase) functioning as a non-core (oblique) argument or adjunct
case - case marking	Used for any case-marking element which is treated as a separate syntactic word (including prepositions, postpositions, and clitic case markers)
conj - conjunct	The relation between two elements connected by a coordinating conjunction, such as and, or, etc.
flat	Is one of three relations for multiword expressions multiword expressions (MWEs) in UD.
advmod - adverbial modifier	A (non-clausal) adverb or adverbial phrase that serves to modify a predicate or a modifier word.
cc - coordinating conjunction	The relation between a conjunct and a preceding coordinating conjunction.

Table 3.1: UD dependency description [niv]

rules which we will define in the upcoming subsections.

3.2 Rule based approach

The rule-based approach for intent classification task is an approach that involves defining a system set of rules that can match the patterns in a given sentence or query to a predefined set of intents.
A rule-based system is a type of expert system that utilizes a set of rules, which can be constructed by applying expert knowledge or learning from real-world data [LG].
Rules construction through expert knowledge is domain-dependent, meaning we need an expert in the domain to maintain the rules and build the system. The other data-based approach uses supervised or unsupervised learning techniques to generate rules and attributes of unknown data using the known data instances [Bra07].
The rules can be defined using regular expressions or other pattern-matching techniques, and the system uses these rules to classify the user's intent.
The advantage of a rule-based system is that it can be designed and customized to a specific domain and achieve high accuracy for well-defined patterns. However, it may not perform well when faced with novel or complex patterns, and it requires human expertise

to define and maintain the rules [RLKH].
A combination of predicting rules and generating by domain experts is a hybrid system that can contribute to the advantages of a rules-based system.
One approach to defining rules for intent classification is regular expressions.
A regular expression is a string of letters, numbers, and special symbols that describes one or more search strings [Bha05].
Regular expressions are able to perform a variety of NLP tasks, including intent classification [MYJ18]
Let us define an algorithm for using regular expressions on intent classification.

Algorithm 1 Intent classification using regular expressions on ATIS dataset

```
Define regular expressions for each intent
$flight\_regex \leftarrow r" \wedge (show|flight|flights)\$"$
$airfare\_regex \leftarrow r" \wedge (travel|airfare|go)\$"$
$airline\_regex \leftarrow r" \wedge (which|airline)\$"$
$aircraft\_regex \leftarrow r" \wedge (aircraft|plane|type)\$.*"$
Define a dictionary to map the regular expressions to intents
$regex\_to\_intent \leftarrow []$
$regex\_to\_intent[flight\_regex] \leftarrow "flight"$
$regex\_to\_intent[airfare\_regex] \leftarrow "airfare"$
$regex\_to\_intent[airline\_regex] \leftarrow "airline"$
$regex\_to\_intent[aircraft\_regex] \leftarrow "aircraft"$
function CLASSIFY_INTENT($sentence$)
    forall $regex, intent\ in\ regex\_to\_intent.items()$ do
        if RE.MATCH(regex, sentence) then
            return $intent$
        end
    if no match is found return "empty"
end
```

From the algorithm 1, even though it may seem straightforward, some significant problems with a regular expression approach can make the intent classification task quite difficult. For example:

- Complexity. Regular expressions are often hard to understand because of their terse syntax, and sheer size [EG12].

- Errors. Many regular expressions in repositories and on the web contain faults. Moreover, these faults are often quite subtle and hard to detect [EG12].

- Version Proliferation. Since in repositories, there can be many versions or regular expressions stored for one particular purpose. Therefore, finding or selecting the right one for a specific task is difficult [EG12].

These difficulties lead to problems, especially with precision and recall of the rules [MYJ18].
Considering these difficulties for regular expression, we will consider outcomes from UD graphs to overcome the problem with precision and recall in the case of using regular expressions only.

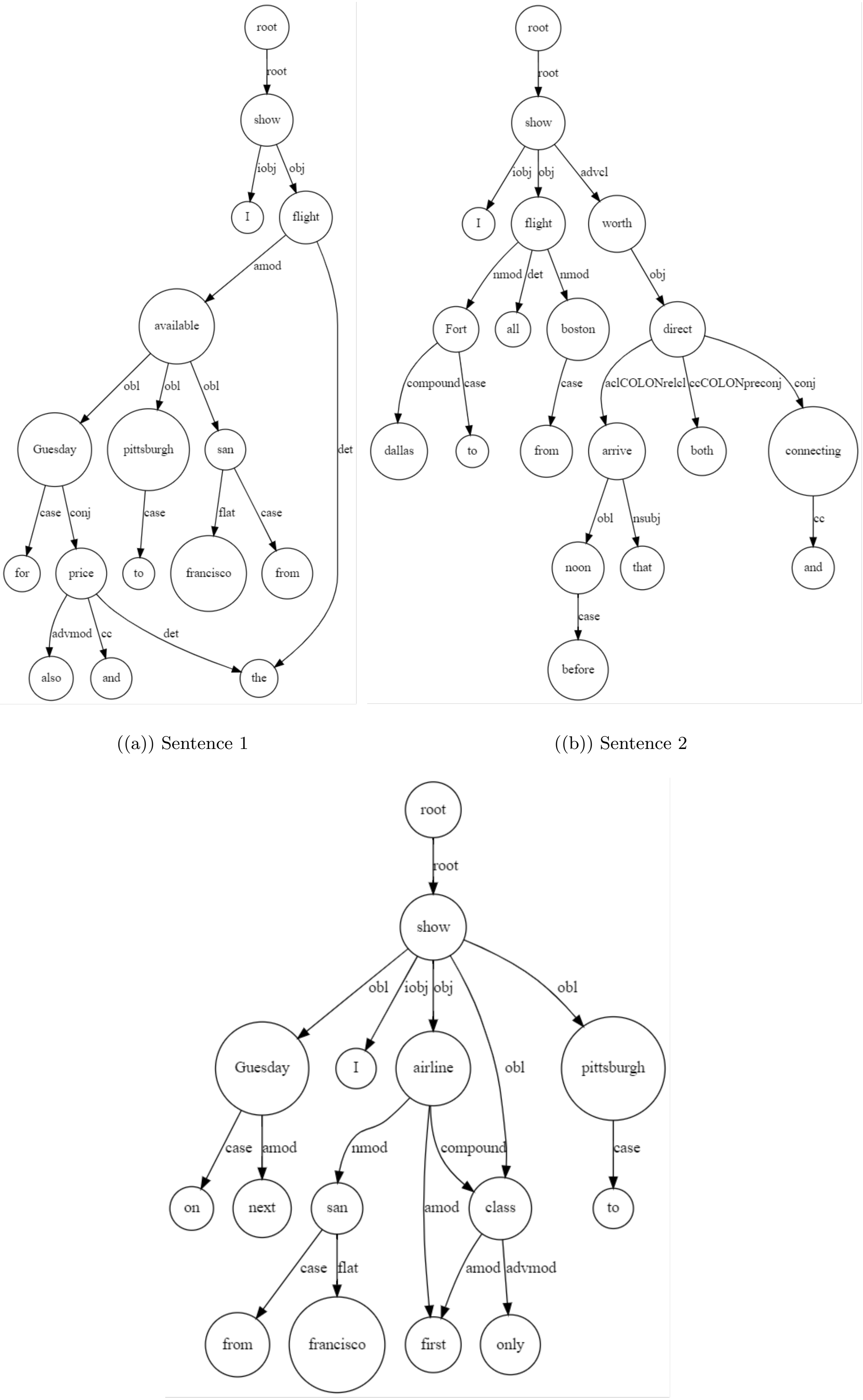
3.3 Graph approach on Rules

UD syntactic graphs help us represent a sentence's grammatical structure in a standardized way.
By using syntactic graphs, relationships between words are made clear, and this can make building regular expressions for intent classification easier.
The syntactic graph representation benefits include information about parts of speech, morphological features, and syntactic dependencies, by which we can identify the key features relevant to a particular intent.
We use these benefits to guide the construction of regular expressions that match the relevant patterns in the sentence.
Using syntactic graphs can also reduce the complexity of the regular expressions needed for intent classification, as the system can focus on the relevant parts of the sentence structure.
Overall, UD syntactic graphs can improve the accuracy and robustness of rule-based systems for intent classification.
Let us take an example of three sentences from ATIS [MS CNTK19] dataset and analyze their syntactic graph representation.

"Show me the flights available from San Francisco to Pittsburgh for Tuesday and also the price [MS CNTK19]." Intent: *"Flight"*

"Show me all flights from Boston to Dallas fort worth both direct and connecting that arrive before noon [MS CNTK19]." Intent: *"Flight"*

"Show me first class airlines from San Francisco to Pittsburgh on next Tuesday first class only [MS CNTK19]." Intent: "Airline"



((a)) Sentence 1

((b)) Sentence 2

((c)) Sentence 3

Figure 3.2: 3 sentences UD syntactic graph representation example

As seen from the sentences, the last sentence refers to the airline, and the first two are about flights. They appear to have similar word tags at first sight, including show, flight, to, and from.

Regular expressions generated solely based on word tags may lead to biased results with issues with precision and recall. Therefore, let us examine the syntactic graph representation using UD in these three cases.

The graph analysis shows that we have the ***show*** word tag in all cases but connected differently in the graph. Also important to mention is that all the sentences start with the same word tag as described in the graph with the root node.

The table below shows the rules for these three sentences and their intents.

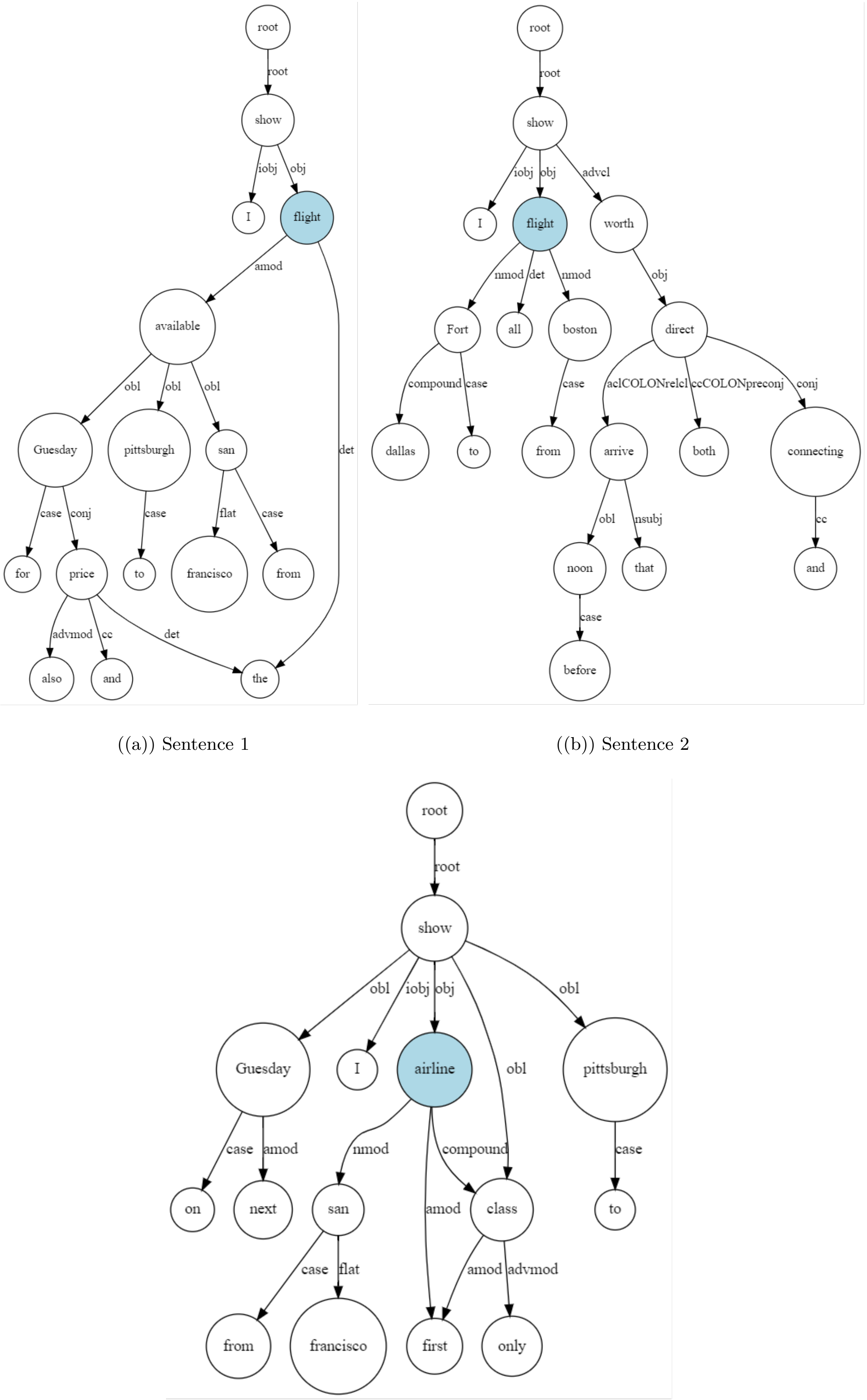
Rule	Intent
(.*/show:iobj(.*/I):obj(.*/airline))	airline
(.*/root:root(.*/show:obj(trip|itinerary|flight|departure)))	flight

Table 3.2: Rules and intents

The table 3.2 shows that a regular expression can be any node connected with show and Flight and Airline intentions. Therefore, we use regular expressions to generalize cases in the dataset.

Then for evaluation, we can see that nodes in the graph turn blue when a rule is fired, indicating that the condition for a particular intent is fulfilled.

Using this approach, we identify different patterns to generate rules using a combination of UD and regular expressions.



((a)) Sentence 1

((b)) Sentence 2

((c)) Sentence 3

Figure 3.3: 3 sentence UD syntactic graph representation rule evaluation example

3.4 Explainable Artificial Intelligence frameworks on interpretable graph features

Building rule-based systems for intent classification is a demanding and time-consuming task requiring much knowledge and data analysis.
Creating rules manually can be biased and might miss out on some essential patterns in the data, making the system unreliable.
We can utilize the power of explainable artificial intelligence (XAI) frameworks to make the process easier by automatically extracting rules from syntactic graph representations and regular expressions.
Kovac et al. introduced a framework called POTATO [KGIR22], which stands for *"exPlainable infOrmation exTrAcTion framewOrk"*, designed to help people create rule-based text classifiers that use graph-based features.
This framework involves humans in the learning process, which is why it is called HITL, which stands for human-in-the-loop, so that humans will have an important impact on classifier performance[KGIR22].
This way, people can build text classifiers by monitoring and contributing to rules creation from the graph representation of text.
POTATO is available on GitHub[gita] and via pip[pypa] by installing the xpotato[pypb] package [KGIR22].
It generates rule suggestions by using graph features to train interpretable machine learning models, where the extracted features of each graph are its connected subgraphs with a maximum of n edges[KGIR22]. Using UD graphs, graph construction consists of a single parser in stanza[QZZ+20].
A system workflow of POTATO is presented in the figure below.

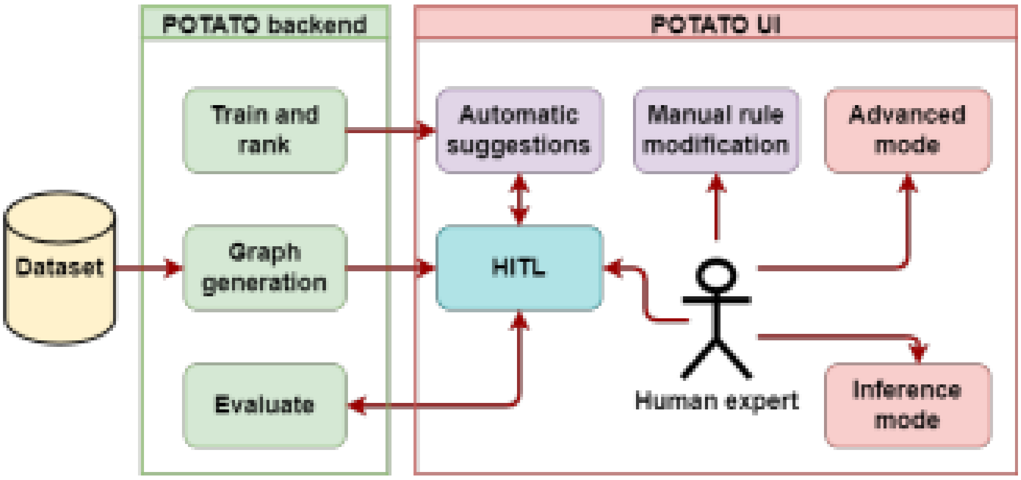
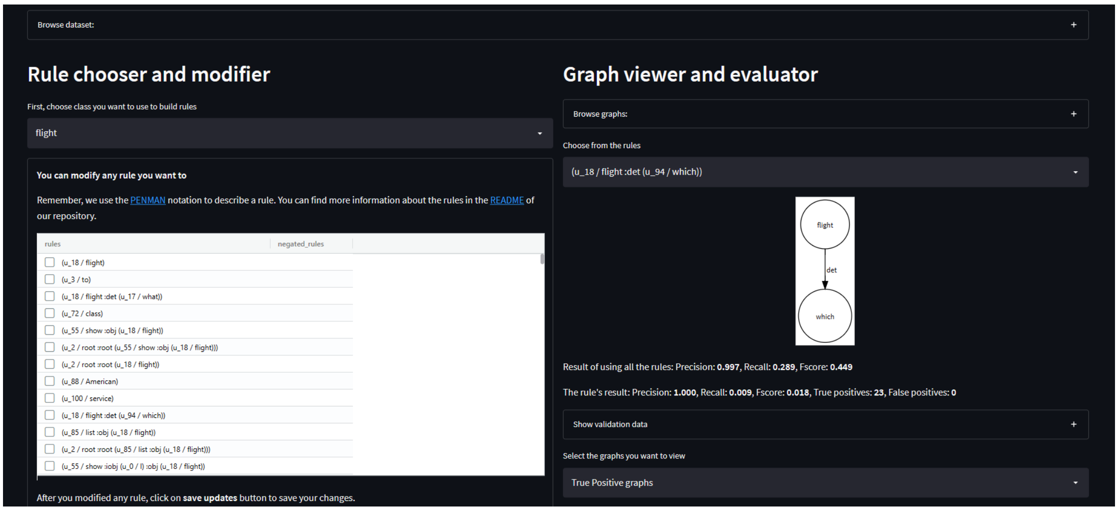


Figure 3.4: System workflow of POTATO[KGIR22]

The process of suggesting and evaluating rules requires truth labels. Therefore, POTATO also offers an advanced option for generating them [KGIR22].
The loading of a dataset as a collection of labeled or unlabeled graphs is a requirement for starting an interface. Any directed graph can be loaded. The DiGraphMatcher class from networkx.algorithms.isomorphism, which implements the vf2 algorithm, can be customized and wrapped by the UD class [FSV01] [KGIR22].
After loading a dataset, the HITL frontend can be launched, and the user is presented with the interface shown in 3.5, which was built using the streamlit[str][KGIR22].



Rule chooser and modifier

First, choose class you want to use to build rules

flight

You can modify any rule you want to

Remember, we use the PENMAN notation to describe a rule. You can find more information about the rules in the README of our repository.

rules	negated_rules
□ (u_18 / flight)	
□ (u_3 / to)	
□ (u_18 / flight :det (u_17 / what))	
□ (u_72 / class)	
□ (u_55 / show :obj (u_18 / flight))	
□ (u_2 / root :root (u_55 / show :obj (u_18 / flight)))	
□ (u_2 / root :root (u_18 / flight))	
□ (u_88 / American)	
□ (u_100 / service)	
□ (u_18 / flight :det (u_94 / which))	
□ (u_85 / list :obj (u_18 / flight))	
□ (u_2 / root :root (u_85 / list :obj (u_18 / flight)))	
□ (u_55 / show :iobj (u_0 / I) :obj (u_18 / flight))	

After you modified any rule, click on **save updates** button to save your changes.

Graph viewer and evaluator

Browse graphs:

Choose from the rules

(u_18 / flight :det (u_94 / which))

Result of using all the rules: Precision: **0.997**, Recall: **0.289**, Fscore: **0.449**

The rule's result: Precision: **1.000**, Recall: **0.009**, Fscore: **0.018**, True positives: **23**, False positives: **0**

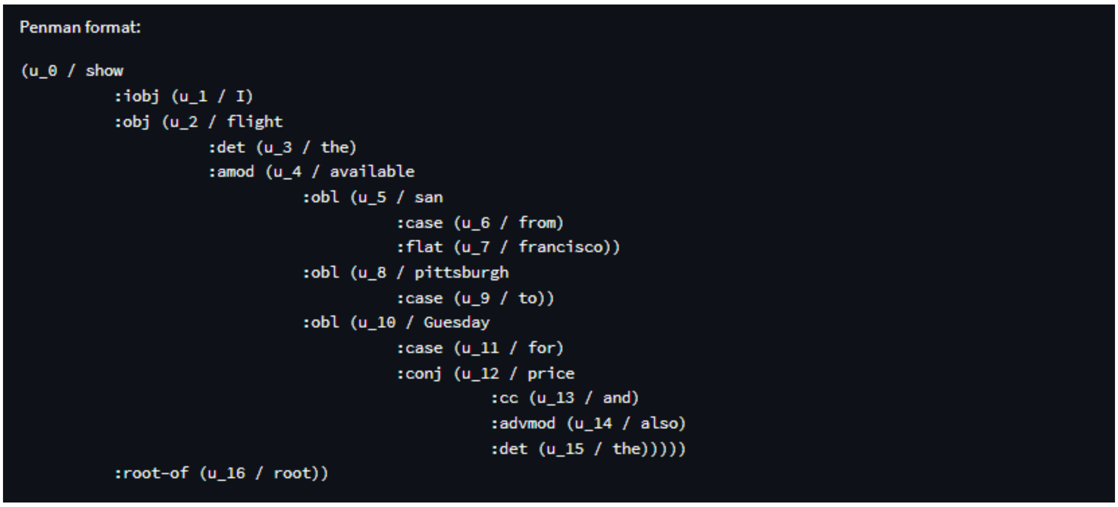
Show validation data

Select the graphs you want to view

True Positive graphs

Figure 3.5: Main page of POTATO User Interface (UI) framework

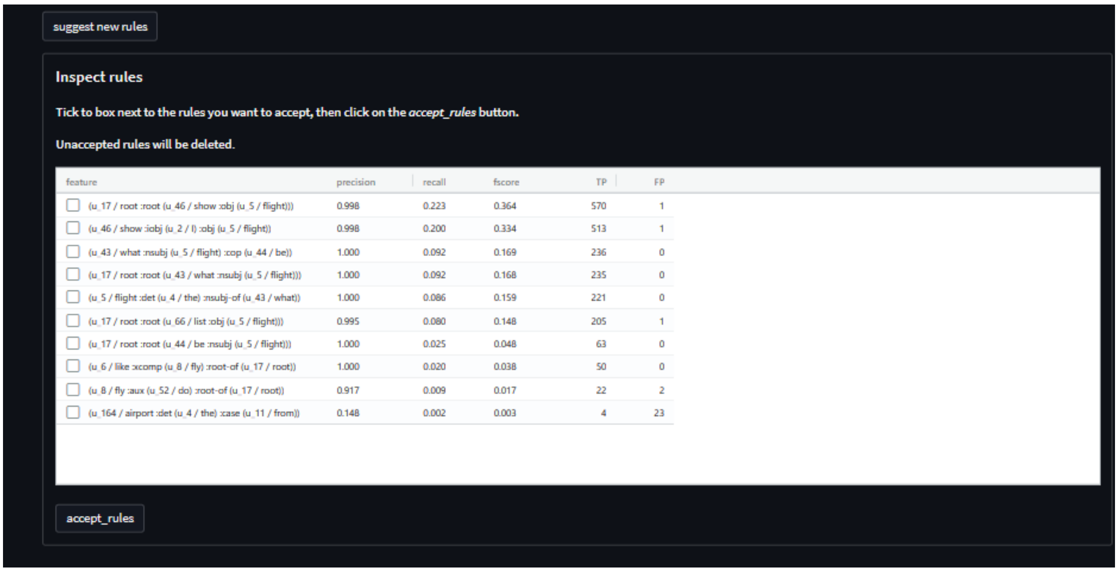
The dataset browser displays each row's text, graph, and label. Furthermore, the viewer renders graphs using the graphviz library[GN00] and provides the PENMAN[Goo20] notation, which the user can copy to edit rules quickly.
Figure 3.6 shows an example of PENMAN notation of graphs representation.
Figure 3.5 depicts how users can select the class to work on and the rules built for each class from a list.
On the training and validation datasets, rules can be viewed and evaluated [KGIR22].
Users can examine true positive, false positive, and false negative examples to determine each rule's correct and incorrect predictions[KGIR22].
Potato also offers the option of suggesting new rules. The figure below shows an example of suggesting rules on ATIS dataset for intent 'flight.'
Figure 3.7 shows that the precision, recall, f-score, true positive, and false positive cases are the base for ranking the suggestions from POTATO.



Penman format:

```
(u_0 / show
          :iobj (u_1 / I)
          :obj (u_2 / flight
                    :det (u_3 / the)
                    :amod (u_4 / available
                              :obl (u_5 / san
                                        :case (u_6 / from)
                                        :flat (u_7 / francisco))
                              :obl (u_8 / pittsburgh
                                        :case (u_9 / to))
                              :obl (u_10 / Guesday
                                        :case (u_11 / for)
                                        :conj (u_12 / price
                                                  :cc (u_13 / and)
                                                  :advmod (u_14 / also)
                                                  :det (u_15 / the)))))
          :root-of (u_16 / root))
```

Figure 3.6: PENMAN notation. Example from figure 3.3 sentence 1



Inspect rules

Tick to box next to the rules you want to accept, then click on the *accept_rules* button.

Unaccepted rules will be deleted.

feature	precision	recall	fscore	TP	FP
□ (u_17 / root :root (u_46 / show :obj (u_5 / flight)))	0.998	0.223	0.364	570	1
□ (u_46 / show :iobj (u_2 / I) :obj (u_5 / flight))	0.998	0.200	0.334	513	1
□ (u_43 / what :nsubj (u_5 / flight) :cop (u_44 / be))	1.000	0.092	0.169	236	0
□ (u_17 / root :root (u_43 / what :nsubj (u_5 / flight)))	1.000	0.092	0.168	235	0
□ (u_5 / flight :det (u_4 / the) :nsubj-of (u_43 / what))	1.000	0.086	0.159	221	0
□ (u_17 / root :root (u_66 / list :obj (u_5 / flight)))	0.995	0.080	0.148	205	1
□ (u_17 / root :root (u_44 / be :nsubj (u_5 / flight)))	1.000	0.025	0.048	63	0
□ (u_6 / like :xcomp (u_8 / fly) :root-of (u_17 / root))	1.000	0.020	0.038	50	0
□ (u_8 / fly :aux (u_52 / do) :root-of (u_17 / root))	0.917	0.009	0.017	22	2
□ (u_164 / airport :det (u_4 / the) :case (u_11 / from))	0.148	0.002	0.003	4	23

Figure 3.7: POTATO rules suggestion.

POTATO uses decision trees trained with the scikit-learn [PVG^{+}12] and evaluates subgraphs by their Gini coefficients [KGIR22].
In addition to using a machine learning approach with a decision tree, POTATO also offers another option for ranking subgraphs.
This method involves counting the number of positive and negative examples in the training data that contain each subgraph as a feature and calculating the difference between the number of true positive and false positive decisions that would result from classifying input sentences based on the presence of this pattern only [KGIR22].

UI option of POTATO is very user-friendly, but depending on the dataset, it may take time to process the suggestions, especially the parsing part of the graphs.
Starting from this conclusion, we can directly use the backend of POTATO.
The main module of the xpotato package[pypb], which is the backend of POTATO, interfaces with the tuw_nlp[gitb][RLKH] module, the scikitlearn library[PVG+12] for training and inspecting decision trees, and the scikit-criteria[CLZ16] package for feature ranking to implement core functionalities [KGIR22].
In the table 3.3 below, we can see the rules generated from ML on POTATO automatically. From these suggestions, there is quite a large number of false negative cases which indicates a lower recall. Giving us the indication that using POTATO suggested cases only can lead to problems with recall.

Rules	**Precision**	**Recall**	**F-score**	**Samples**	**TP**	**FP**	**FN**
$(u\_55/show\ obj\ (u\_18/flight))$	0.998758	0.219313	0.359651	3666	804	1	1699
$(u\_2/root\ root\ (u\_55/show\ obj(u\_18/flight)))$	0.998756	0.21904	0.359284	3666	803	1	1699
$(u\_55/show\ iobj\ (u\_0/I)\ obj\ (u\_18/flight))$	0.998609	0.195854	0.32748	3666	718	1	1699
$(u\_18/flight\ det\ (u\_45/a))$	0.950909	0.142662	0.248102	3666	523	1	1699
$(u\_2/root\ root\ (u\_17/what\ nsubj\ (u\_18/flight)))$	1	0.090289	0.165624	3666	331	27	1699
$(u\_2/root\ root\ (u\_85/list\ obj(u\_18/flight)))$	0.996795	0.084834	0.15636	3666	331	0	1699

Table 3.3: Rules from POTATO

3.5 Expertise impact on Rules extraction from Graphs

In the previous section, we have seen the benefits of using XAI frameworks for building rules from syntactic text representation graphs. However, although we have good precision values, the f-score value stands low. One indication of this is the low recall.

In machine learning, recall measures a model's ability to identify all relevant instances in a dataset. Specifically, recall is the proportion of true positive instances the model correctly identified out of all positive instances.

A low recall tells us that the model is either more sensitive to the relevant features in the data or that the model's decision point is set too high, giving us results in too many true positives cases wrongly classified as false negatives cases.

This indicates that involving human expertise is very crucial.

Human experts' expertise in using rule-based systems like POTATO is significant.

Rule-based systems depend on human expertise to create and refine the rules for intent classification.

In addition, defining rules can benefit from human expert domain expertise and intuition, making the system's text classification more precise and efficient.

In the case of XAI frameworks such as POTATO[KGIR22], the system's "human-in-the-loop" context provides users the possibility to determine and refine the rules actively.

This possibility ensures that the system is adjusting to the specific needs of the users and the domain in which we are using the system.

Human expertise help to identify patterns and features in the text that cannot be immediately apparent by the system on its own, guiding to better overall performance.

From the table 3.4, we can see the difference, especially for the decrease in False Negative (FN) cases and the increase of True Positive (TP), including an increase for some of the rules on recall and also on f-score. We created our Rule Based System (RBS) from rules generated from POTATO, including human expertise, to reduce the number of FN.

The table 3.5 below shows the rules we used for Flight and Airline intent classes in the RBS. You can find the rest of the rules for all the intent classes in the rule-based system in Appendix A.1.

Rules	**Precision**	**Recall**	**F-score**	**Samples**	**TP**	**FP**	**FN**
$(u\_55/show\ obj(trip\vert itinerary\vert flight\vert departure))$	0.99877	0.221495	0.362581	3666	812	1	474
$(u\_2/root\ root(u\_55/show\ obj(trip\vert itinerary\vert flight\vert departure)))$	0.998768	0.221222	0.362215	3666	811	1	474
$(u\_55/show\ iobj(we\vert I)\ obj(trip\vert itinerary\vert flight))$	0.998621	0.19749	0.329765	3666	724	1	474
$(u\_18/flight\ det(that\vert which\vert all\vert any\vert the\vert milwaukee\vert what\vert a))$	0.968864	0.653573	0.780583	3666	2396	77	474
$(u\_2/root\ root(.*))$	0.950604	0.236225	0.378414	3666	866	45	474
$(u\_2/root\ root(u\_85/list\ obj(landing\vert takeoff\vert all\vert trip\vert flight)))$	0.996894	0.087561	0.160983	3666	321	1	474

Table 3.4: Rules from POTATO when a human expert is involved

<table>
<tr><th>Rules</th><th>Intent</th></tr>
<tr><td>[[’(u_55/show:obj(trip|itinerary|flight|departure))’], [], "flight"],</td><td>Flight</td></tr>
<tr><td>[[’(u_2/root:root(u_55/show:obj(trip|itinerary|flight|departure)))’], [], "flight"],</td><td>Flight</td></tr>
<tr><td>[[’(u_55/show:iobj(we|I):obj(trip|itinerary|flight))’], [], "flight"],</td><td>Flight</td></tr>
<tr><td>[[’(u_18/flight:det(that|which|all|any|the|milwaukee|what|a))’], [], "flight"],</td><td>Flight</td></tr>
<tr><td>[[’(u_2/root:root(request|meal|X10|florida|atlanta|interested|chicago|start|miami|want|wish|louis|return|connect|sfo|information|live|newark|need|thank|wednesday|how| seattle|flight|arrive|petersburg|make|we|to|sorry|when|charlotte|listing|carry|toronto|vegas|display|philadelphia))’], [], "flight"],</td><td>Flight</td></tr>
<tr><td>[[’(u_2/root:root(u_85/list:obj(landing|takeoff|all|trip|flight)))’], [], "flight"],</td><td>Flight</td></tr>
<tr><td>[[’(u_3/root:root (u_50/show:obj(airfare|ticket|cost|fare|price)))’], [], ’airfare’],</td><td>Airfare</td></tr>
<tr><td>[[’(u_18/flight:case(for|of|on):nmod-of(cost|fare|price|airfare))’], [], ’airfare’],</td><td>Airfare</td></tr>
<tr><td>[[’(u_159/cost:xcomp(go|fly|travel):aux(u_71/do))’], [], ’airfare’],</td><td>Airfare</td></tr>
<tr><td>[[’(u_135/ticket:det(what|the|a))’], [], ’airfare’],</td><td>Airfare</td></tr>
<tr><td>[[’(u_15/root:root(much|airfare|ticket|fare|price))’], [], ’airfare’],</td><td>Airfare</td></tr>
<tr><td>[[’(u_15/root:root(u_49/show:obj(airfare|ticket|cost|fare|price)))’], [], ’airfare’],</td><td>Airfare</td></tr>
<tr><td>[[’(u_49/show:obj(fare|cost|ticket|price))’], [], ’airfare’],</td><td>Airfare</td></tr>
<tr><td>[[’(u_12/fare:compound(thrift|cost|air|trip))’], [], ’airfare’],</td><td>Airfare</td></tr>
<tr><td>[[’(u_14/fare:det(u_13/the):nsubj-of(u_41/what))’], [], ’airfare’],</td><td>Airfare</td></tr>
<tr><td>[[’(u_41/what:nsubj(fare|ticket|price|airfare):cop(u_32/be))’], [], ’airfare’],</td><td>Airfare</td></tr>
<tr><td>[[’(u_3/root:root(u_41/what:nsubj(fare|ticket|price|airfare)))’], [], ’airfare’],</td><td>Airfare</td></tr>
<tr><td>[[’(u_50/show:iobj(we|I):obj(fare|cost|ticket|price))’], [], ’airfare’],</td><td>Airfare</td></tr>
<tr><td>[[’(u_159/cost:xcomp(go|fly|travel):aux(u_71/do))’], [], ’airfare’],</td><td>Airfare</td></tr>
<tr><td>[[’(u_5/fly:mark(u_4/to):xcomp-of(u_159/cost))’], [], ’airfare’],</td><td>Airfare</td></tr>
<tr><td>[[’(u_3/root:root(u_159/cost:xcomp(go|take|fly|travel)))’], [], ’airfare’],</td><td>Airfare</td></tr>
<tr><td>[[’(u_3/root:root(much|airfare|ticket|fare|price))’], [], ’airfare’],</td><td>Airfare</td></tr>
<tr><td>[[’(u_80/price:nmod(ea|we|morning|economy|ticket|class|air|seat|flight|fare))’], [], ’airfare’],</td><td>Airfare</td></tr>
<tr><td>[[’(u_41/what:nsubj(fare|ticket|price|airfare))’], [], ’airfare’],</td><td>Airfare</td></tr>
<tr><td>[[’(u_12/fare:det(u_13/the):nsubj-of(u_41/what))’], [], ’airfare’],</td><td>Airfare</td></tr>
</table>

Table 3.5: Rules used on Rule-based system. Cases of Flight and Airfare as most popular intents on the ATIS[MS CNTK19] dataset.

CHAPTER 4

Intent classification through Machine Learning

Intent classification is an essential task in natural language processing that involves understanding the meaning or purpose of a user's input.
While rule-based systems have traditionally been used for this task, they struggle to handle the complexities and variations of natural language.
To address this, machine learning techniques such as SVM and BERT have been applied to intent classification with promising results, as they can identify intricate patterns in the data.
This master thesis presents a comparison of the performance of SVM and BERT approaches with a rule-based syntactic graph system and explores the feasibility of a hybrid system that integrates SVM and rule-based methods.
The study seeks to determine whether this hybrid system can improve the accuracy and robustness of intent classification by utilizing the strengths of both approaches.
This chapter will present the SVM and BERT approach to user intent classification of the ATIS[MS CNTK19] dataset.

4.1 Support Vector Machines - SVM

The SVM algorithm is based on constructing an optimal hyperplane, which we use to classify linearly separable patterns[Pra12].
From the set of hyperplanes, an optimal hyperplane is chosen for classifying patterns that maximize the hyperplane's margin[Pra12].

SVM is a traditional machine learning algorithm that works well for linearly separable data.
From figure 4.1, we have an example of the logic of SVM algorithm, which does the linear

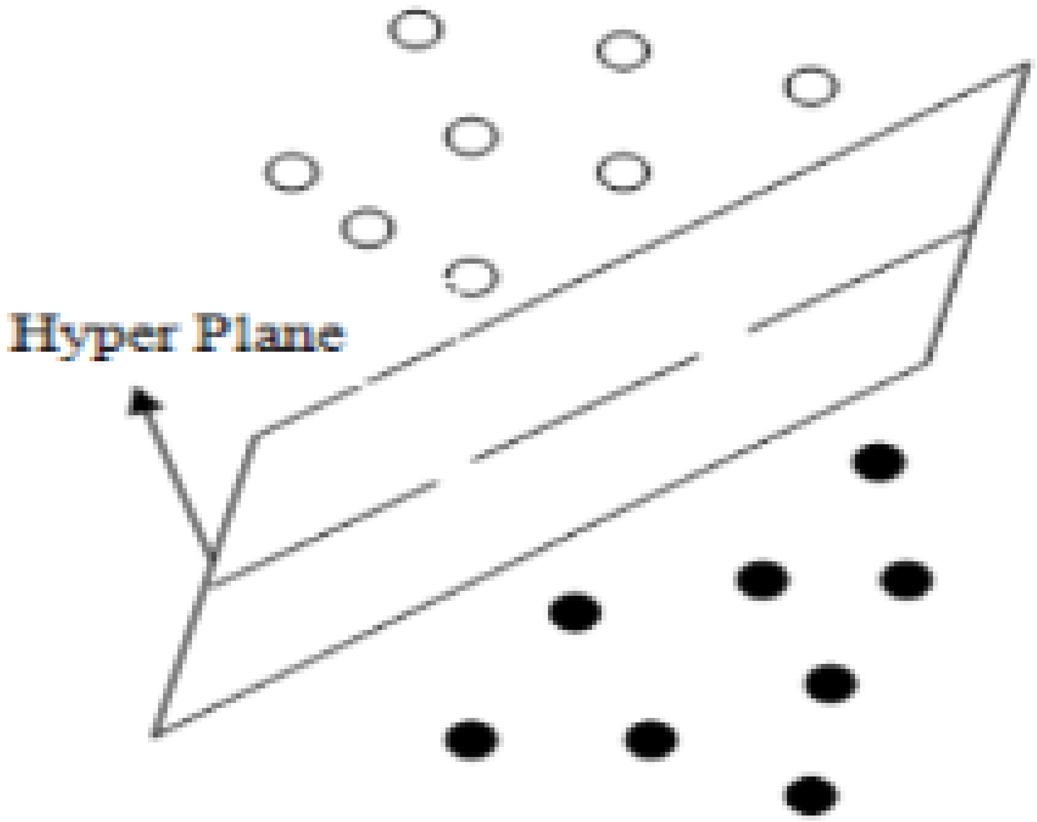


Figure 4.1: SVM hyperplane [Pra12].

separation between two classes.
SVM has been successful in NLP tasks, particularly when the number of features is small and the classes are well-defined[Kec05]. However, SVM can struggle with complex, non-linear data and may not perform well when the number of features is large.
Also, SVM, while used on intent classification, provides very good precision with a low recall [IKT05].
But why should we use SVM?
SVM compared to other ML supervised algorithms has some advantages:

Advantages	**Disatvantages**
• It gives good results even if there needs to be more information about the data. It also works well with unstructured data. • Solves complex problems with a convenient kernel solution function. • Relatively good scaling of highdimensional data.	• It is not easy to choose the appropriate kernel solution function. • Training time is extended when using large data sets. • It may be challenging to interpret and understand because of problems caused by personal factors and the weights of variables. • The weights of the variables are not constant. Thus the contribution of each variable to the output is variant.

Table 4.1: SVM advantages and disadvantages [Joa99]

The table 4.1 shows the advantages and disadvantages of SVM.
In terms of weaknesses, may require more human input to fine-tune the model's parameters and features, which can be time-consuming and require expertise, especially on setting the weights parameter[Kec05].

4.1.1 Experiment setup for SVM

This section presents the experiment setup for SVM model prediction on the ATIS database.
In the algorithm 3 presented below, we have detailed each step on how the model and preprocessing of the dataset was performed to do intent classification on the ATIS dataset.

Algorithm 3 SVM algorithm for intent classification.

```
Input: Dataframe $df$, spaCy model $nlp$, SVM model hyperparameter $C$
Extract feature and label data from $df$: $x\_train\_SVM$ ← drop the 'label' column of $df$, $y\_train\_SVM$ ← 'label' column of $df'Set$ embedding_dim ← length of spaCy model's vectors
Convert sentences in $x\_train\_SVM$ to list of strings: $sen\_train$ ← 'text' column of $x\_train\_SVM$ as list
Convert labels in $y\_train\_SVM$ to list of integers: $labels\_train$ ← 'label' column of $y\_train\_SVM$ as list, and encode using LabelEncoder
Generate feature matrix using spaCy vectorization: $train\_X$ ← call $encode\_sentences$ function with $sen\_train$ as input
Train an SVM model on the feature matrix and encoded labels: $clf$ ← instantiate an SVM model with hyperparameter $C$, $model$ ← fit $train\_X$ and $labels\_train$ to $clf$
Generate predicted labels for the training set: $y\_true\_SVM, y\_pred\_SVM$ ← $labels\_train$, call SVM model's $predict$ function with $train\_X$ as input
Print classification report for the predicted labels
Output: SVM model $model$ =0
```

Before training the SVM model, since we are dealing with an NLP problem, we first need to preprocess the data.

The sentences have been preprocessed using spacy[spa], which is a natural processing python package.

We vectorize sentences using spacy by passing each token with an NLP object.

Besides vectorization, we need to label and encode the labels we want to predict. Label encoding is performed using sklearn.preprocessing[lab] package from python. It encodes each label intent with a number since SVM works with numerical inputs to do the classification.

For the SVM model with ATIS data, we have used the package sklearn.svm[sci] from scikit-learn[PVG$^+$12] in python.

The benefit of the SVM from scikit-learn is that the model comes with most of the parameters in default. The only parameter which does not come with a default state is the regularization parameter C.

A regularization parameter is a positive number that tells us how much we want to avoid miss classification on training data [sci].

We are interested in soft margin SVM, so we have set the regularisation parameter to 1.

Another important parameter is the SVM kernel, and we are using Radial Basis Function (RBF) kernel. This kernel is the default kernel while using SVM from the scikit-learn python package.

The kernel is important in SVM because it takes the data as input and is responsible for handling them in a proper format for linear separation.

The RBF kernel maps high dimensional space data into lower dimensional to perform SVM classification.

SVM has a group of meta parameters[sci], but our focus is on the regularization parameter and kernel. The rest of the parameters we used are the default once from the scikit-learn package.
After the model has been defined, we train the model using a train set, and afterward, the prediction from the model is performed on the validation set and, later on, the test set of the data, which will be described in more detail on the chapter *Results*.
Table 4.2 presents the distribution of sentences on the ATIS dataset.

Dataset	**Number of Sentences**
Train set	3951
Validation set	988
Train and Validation set	4939
Test set	870

Table 4.2: Train, validation and test set.

4.2 BERT

BERT(Bidirectional Encoder Representations from Transformers)[DCLT18] is a deep learning pre-trained model.
BERT is intended to jointly adjust the left and right background in all layers to pre-train deep bidirectional representations from an unlabeled text. [DCLT18].
An advantage of the BERT algorithm is that we can use a pre-tuned BERT model in that it can be fine-tuned with just one additional output layer[DCLT18].
In figure 4.2, we can see the architecture of BERT in pre-training and fine-tuning cases.

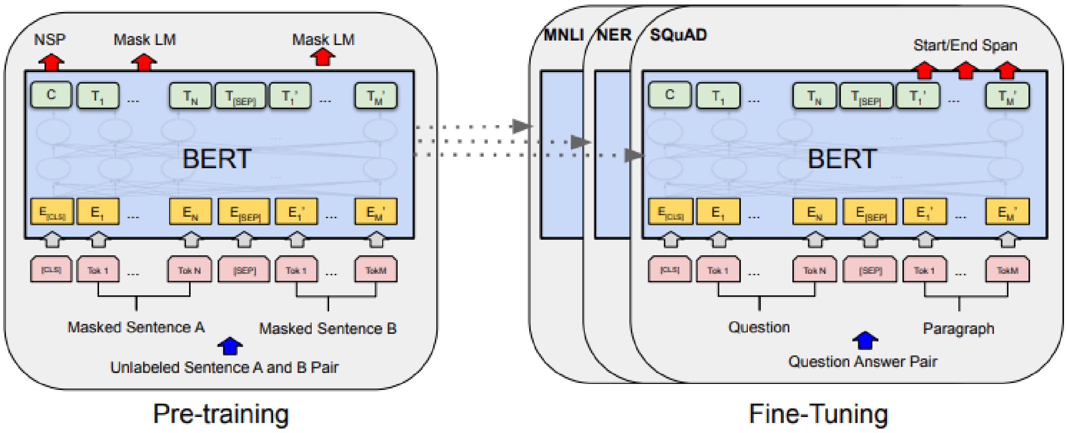


Figure 4.2: BERT model architecture [DCLT18].

BERT performs exceptionally well in intent classification, even when many features or classes exist.
One of the main advantages of BERT is its ability to capture contextual information and understand the meaning behind words in a sentence, leading to more accurate predictions.
On the other hand, BERT can be computationally expensive and requires a significant amount of training data to perform well.
Another disadvantage of the BERT model is the "black box" effect that we have on the network layers it uses and how they individually generate the training process for the model.
The black box effect can lead to difficulties in identifying the logic of how BERT identifies patterns for classification and if there are any hidden biases present.

4.2.1 Experiment setup for BERT

This section will define the experiment setup for pre-trained BERT on the ATIS dataset.
For our experiment, we initially loaded the dataset on pandas[WM10] data frame, and labels are converted to numerical values using *"preprocessing.LabelEncoder()"*[lab] function in python.

The second step is to extract the labels and text from the data frame and convert them to arrays.
Until now, we presented a general data preprocessing for NLP tasks. Below we can see the algorithm we used in the BERT case.

Algorithm 4 Pre-trained BERT Algorithm

```
Pre-trained BERT model $M$, input text $X$ Encoded text $E$
BERTEncode($M$, $X$) **Data:** BERT tokenizer $T$
$tokens \leftarrow T.tokenize(X)$
  $input\_ids \leftarrow T.convert\_tokens\_to\_ids(tokens)$
  $attention\_mask \leftarrow T.create\_attention\_mask(input\_ids)$
  $inputs \leftarrow \{input\_ids, attention\_mask\}$
  $output \leftarrow M(inputs)$
  $E \leftarrow output[0]$ ;                    // Get encoded text from model output
**return** $E$
```

As we can see from the algorithm 4, we define a function that takes a pre-trained BERT model and input text as input and returns the encoded text.
We use tokenizer T to tokenize the input text, convert the tokens to IDs, create an attention mask, and pass the resulting inputs to the BERT model. This tokenizer is a pre-trained BERT tokenizer from the hugging face transformers library[WDS+19].
The token IDs, attention masks, and labels are converted to PyTorch tensors[PGM+19] and split into training and validation sets.
DataLoaders are created for the training and validation sets using PyTorch's DataLoader[PGM+19] class, with a specified batch size and random/sequential sampling.
We initialize the optimizer with the recommended learning rate for BERT fine-tuning *"(3e-5)"*.
We use a specific number of epochs and a training loop consisting of forward and backward passes, optimizer updates, and loss calculations for the model.
Stopping criteria of the training model is a significant step in BERT cases.
Therefore, we need to monitor the evaluation and training loss function.
For example, when validation loss decreases from one epoch to another during the training step, we have an indication that the model is not overfitting or underfitting. Also, while the validation loss starts to become constant or slightly increase, we have a trained model indicating we can stop the training process.
The table 4.3 below shows the output of our BERT model training process. Again, we see a drastic drop in training loss and a slight increase in the validation loss, indicating the stopping criteria of the training process.

After completing these steps, we extract and return the encoded text from the model output.

Epoch: 50%, 1/2 [21:29<21:29, 1289.88s/it]	
Train loss:	0.6158
Validation loss:	0.0373
Validation Precision:	1.0000
Validation Recall:	0.8182
Validation Specificity:	1.0000
Epoch: 100% 2/2 [59:42<00:00, 1791.33s/it]	
Train loss:	0.1166
Validation loss:	0.0444
Validation Precision:	0.9375
Validation Recall:	1.0000
Validation Specificity:	0.9600

Table 4.3: Train BERT model

4.3 SVM and Rule-Based Hybrid system

In the algorithm 5 below, we have a merge on predictions from the SVM approach and rule-based systems.
This approach contributes to controlling and eliminating any hidden biases.
SVM machine learning approach in an unbalanced dataset can perform hidden biases as for the rules-based systems, the rules are defined by human expertise, so this way, by merging these two approaches, we eliminate any unwanted hidden biases.

Algorithm 5 Hybrid prediction algorithm

Input: *df_rules_pred*: DataFrame containing predicted labels by a rule-based classifier, *df*: DataFrame containing true labels, *predictions_test*: List of predicted labels by a machine learning model-SVM

Output: List of hybrid predictions

```
for number in range(len(df_rules_pred)) do
    if df_rules_pred['Predicted label'].values[number] == ' ' then
        df_rules_pred['Hybrid Prediction'].values[number] = predictions_test[number];
    end
    else
        df_rules_pred['Hybrid Prediction'].values[number] = df_rules_pred['Predicted
         label'].values[number];
    end
end
return df_rules_pred['Hybrid Prediction'].tolist();
```

From the algorithm 5, we have *df_rules_pred* containing labels from a rule-based

classifier.
True labels are set under *df*. On the *predictions_test*, we have predicted labels from the SVM machine learning model.
Based on the algorithm, if we have a rule regarding the entry we are willing to classify, we take the response from rule-based systems. However, if no rule is defined for this case, we make the classification regarding SVM.

CHAPTER 5

Results

Our evaluation of the results takes part in quantitative and qualitative analysis.
With quantitative analysis, we will present the graphical representation and numerical representation of our experiments.
Including dataset distribution, classification reports for user intent classification, and comparisons between rule-based approach, SVM, BERT, and hybrid models.
Qualitative analysis also plays a crucial role in our evaluations.
With qualitative analysis, we will analyze specific case comparisons between models to conclude findings afterward.

5.1 Quantitative evaluation

This section presents a quantitative analysis and evaluation of the results. Let us first present the data distribution of the dataset.
In the figure 5.1 above, we see the distribution of the intent label of the ATIS dataset.
Figure 5.1 shows that we are working with an unbalanced dataset. Meaning that intent flight is dominating the intent class.
The unbalances of data distribution on our target class can lead to challenges for machine learning models because they can develop biased predictions towards the majority class, making it difficult to predict the minority classes accurately.
Another vital metric on text classification with rule-based systems is the frequency of words on the dataset.
Since we are defining rules based on regular expression logic, we need to be cautious that a word might be present in two or more intents, so while defining the rule, we should not exclude it to create any biased decision from rules.

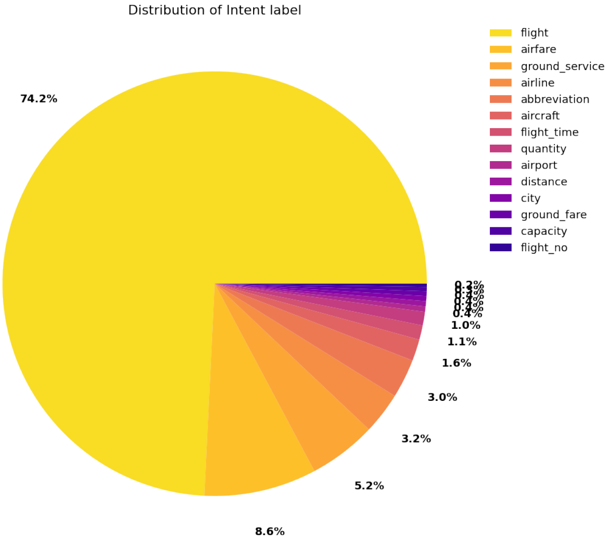


Figure 5.1: Label distribution on ATIS dataset.

The figure 5.2 below shows the distribution of the top-used words on the dataset.

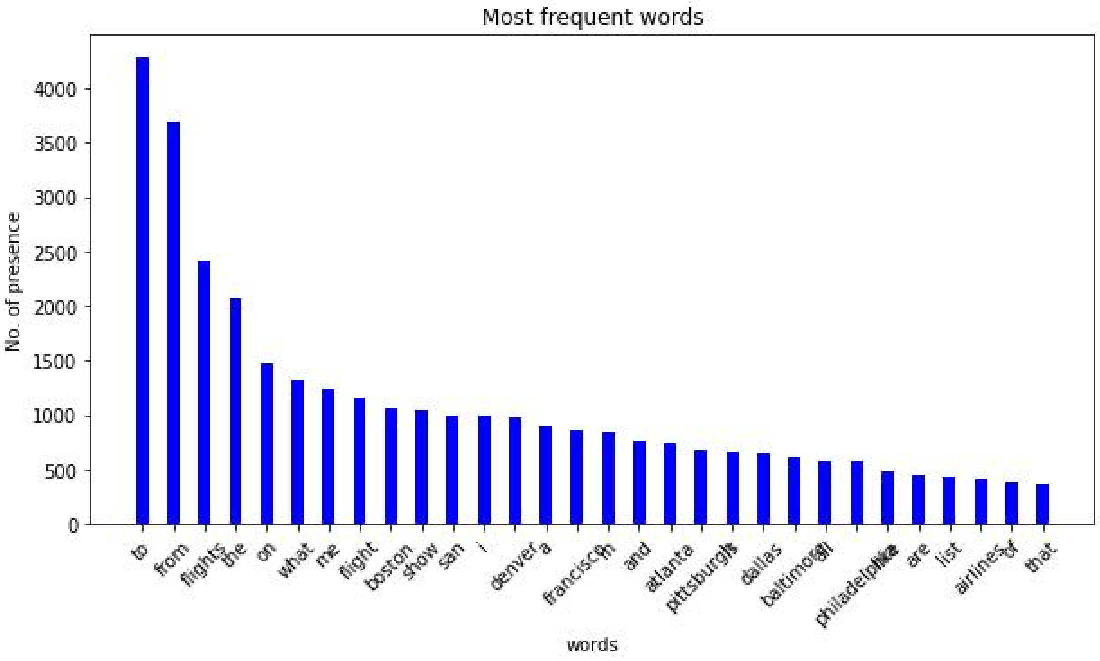


Figure 5.2: Top most frequent words.

Regarding evaluating the results from models, we anticipate comparing the precision, recall, and F1 score as accuracy evaluation metrics of ML, DL, rule-based graph, and hybrid model approaches.

We will base the evaluation of the results on intent classification F1 score, precision, and recall [YA20].

F1 score, or the harmonic mean between precision and recall, is used as a statistical measure to rate performance [Sas07]. Evaluating these parameters of the algorithms will help us answer the research questions and define our conclusions.

Our dataset includes a train data set that we use to generate rules using POTATO and train ML and DL models.

The validation set comes next, where we choose the best parameters for the models and rules, and the test set, which contains unobserved data for the models and rules, is where we see how well they perform on unobserved data.

From the table 5.1 below, we can see the measurement of precision, recall, and f1-score but also the accuracy of our models on the validation set.

Since we are dealing with an unbalanced dataset, we have also considered macro-averaging instead of only micro-averaging.

Micro-averaging involves calculating the overall performance metric by considering the total number of true positives, false positives, and false negatives across all classes [MS].

In macro averaging, we calculate the performance metric for each class separately and then use the average of those metrics [MS].

While macro-averaging gives equal weight to all classes, micro-averaging gives more weight to the performance of the majority of classes [MS].

Table 5.1: Validation data. Precision, Recall and f1-score of classification models.

	SVM			**BERT**			**RBS**			**Hybrid SVM-RBS**		
	precision	*recall*	*f1-score*	*precision*	*recall*	*f1-score*	*precision*	*recall*	*f1-score*	*precision*	*recall*	*f1-score*
macro avg	82.12	51.69	59.76	88.84	79.88	81.51	96.96	90.53	93.43	99.00	86.65	91.58
weighted avg	91.34	92.45	90.97	99.22	99.15	99.11	96.72	85.28	90.48	97.81	97.79	97.68
accuracy	**92.45**			**99.15**			**85.28**			**97.79**		

We see from the table 5.1 that the difference in SVM precision between macro averaging and weighted averaging differs by approximately 10% and which is not the case for other approaches.

Proving that weighting the classes on unbalanced datasets is important to get significant results.

From the table 5.1, we interpret that intent classification was as follows, taking into consideration weighted averaging:

1. **SVM**: Correctly classified 92.45% of the intents on the validation set. Out of them, we have a precision of 91.34% and a recall of 92.45%. The F1 score of 90.97%

indicates a good balance between recall and precision.

2. **BERT**: Correctly classified 99.15% of the intents on the validation set. Out of them, we have a precision of 99.22% and a recall of 99.15%. The F1 score of 99.11% indicates a better balance between recall and precision than SVM.

3. **RBS**: Correctly classified 85.28% of the intents on the validation set. Out of them, we have a precision of 96.72% and a recall of 85.28%. The F1 score of 90.48% indicates a good balance between recall and precision.

4. **Hybrid SVM-Rule Based System (HYBRID SVM-RBS)**: Correctly classified 97.79% of the intents on the validation set. Out of them, we have a precision of 97.81% and a recall of 97.79%. The F1 score of 97.68% indicates a good balance between recall and precision, similar to BERT.

Out of these outcomes, we see that BERT and HYBRID SVM-RBS models have the best and very similar results on the validation set.
Now that we have the evaluation of models on the validation set. It is important to see how the models performed on each intent class.
The table 5.2 below shows the classification report on each intent class of the validation set.

Table 5.2: Validation data. Precision, Recall, and f1-score on each intent, including all classification models.

	SVM			**BERT**			**RBS**			**Hybrid SVM-RBS**		
	precision	*recall*	*f1-score*	*precision*	*recall*	*f1-score*	*precision*	*recall*	*f1-score*	*precision*	*recall*	*f1-score*
abbreviation	91.50	95.24	93.33	99.32	98.64	98.98	96.00	92.00	94.00	98.00	100.0	98.99
aircraft	86.79	56.79	68.66	97.56	98.77	98.16	94.02	77.77	85.13	95.65	81.48	88.00
airfare	92.04	87.47	89.70	97.46	99.76	98.60	96.00	83.00	89.00	98.78	95.51	97.12
airline	94.83	35.03	51.16	94.58	100.0	97.21	97.00	82.00	89.00	97.89	88.54	92.98
airport	100.0	35.00	51.85	95.00	95.00	95.00	100.0	75.00	85.71	100.0	85.00	91.89
capacity	100.0	56.25	72.00	100.0	81.25	89.66	100.0	100.0	100.0	100.0	100.0	100.0
city	100.0	42.11	59.26	100.0	57.89	73.33	100.0	100.0	100.0	100.0	94.74	97.30
distance	100.0	20.00	33.33	100.0	75.00	85.71	100.0	100.0	100.0	100.0	95.00	97.44
flight	92.46	99.29	95.75	99.92	99.86	99.89	96.00	87.00	91.00	97.60	99.78	98.68
flight no	00.00	00.00	00.00	00.00	00.00	00.00	100.0	100.0	100.0	100.0	50.00	66.67
flight time	00.00	00.00	00.00	81.25	96.30	88.14	88.46	85.18	86.79	100.0	57.41	72.94
ground fare	100.0	27.78	43.48	100.0	16.67	28.57	100.0	100.0	100.0	100.0	77.78	87.50
ground service	92.11	96.08	94.05	97.68	99.22	98.44	93.00	95.00	94.00	98.03	97.65	97.84
quantity	100.0	72.55	84.09	80.95	100.0	89.47	97.91	92.15	94.94	100.0	90.20	94.85

The table 5.2 shows that the intent class for *"flight number"* and *"flight time"* shows 0% precision on the SVM model, indicating that SVM did not have any optimistic predictions on these two classes.
They initially indicated that we can have biased results from the SVM model in these two intents. The same situation is with the deep learning BERT model on *"flight number"* intent.
As seen from the table 5.2, the rest intent features have very promising results on all models.

Up to this point, we have seen the evaluation of models from the validation set. From them, BERT and HYBRID SVM-RBS models have top scores on every evaluation metric. But how about the performance of the models on unseen test data?
The table 5.3 below shows the evaluation of the models on the test data.

Table 5.3: Unseen test set data. Precision, Recall, and f1-score of classification models.

	SVM			**BERT**			**RBS**			**Hybrid SVM-RBS**		
	precision	*recall*	*f1-score*	*precision*	*recall*	*f1-score*	*precision*	*recall*	*f1-score*	*precision*	*recall*	*f1-score*
macro avg	72.75	41.39	47.29	72.32	78.37	69.88	96.03	63.23	71.41	90.88	73.54	79.52
weighted avg	89.13	90.62	88.61	96.66	96.67	96.38	93.90	77.82	84.24	94.28	94.94	94.07
accuracy	**90.62**			**96.67**			**77.82**			**94.94**		

From the table 5.3, we can see that:

1. **SVM**: Correctly classified 90.62% of the intents on the validation set. Out of them, we have a precision of 89.13% and a recall of 90.62%. The F1 score of 88.61% indicates a good balance between recall and precision.

2. **BERT**: Correctly classified 96.67% of the intents on the validation set. Out of them, we have a precision of 96.66% and a recall of 96.67%. The F1 score of 96.38% indicates a better balance between recall and precision than SVM.

3. **RBS**: Correctly classified 77.82% of the intents on the validation set. Out of them, we have a precision of 93.90% and a recall of 77.82%. The F1 score of 84.24% indicates a good balance between recall and precision.

4. **HYBRID SVM-RBS**: Correctly classified 94.94% of the intents on the validation set. Out of them, we have a precision of 94.28% and a recall of 94.94%. The F1 score of 94.07% indicates a good balance between recall and precision, similar to BERT.

From the table 5.4 below, we can see the performance of all models for each intent classification class on the test dataset with unseen data for the models.

Table 5.4: Unseen test set data. Precision, Recall and f1-score on each intent including all classification models .

	SVM			**BERT**			**RBS**			**Hybrid SVM-RBS**		
	precision	*recall*	*f1-score*	*precision*	*recall*	*f1-score*	*precision*	*recall*	*f1-score*	*precision*	*recall*	*f1-score*
abbreviation	81.25	86.67	83.87	94.29	100.0	97.06	96.77	90.91	93.75	100.0	100.0	100.0
aircraft	81.82	34.62	48.65	100.0	88.89	94.12	75.00	66.67	70.59	100.0	44.44	61.54
airfare	88.35	84.26	86.26	87.27	100.0	93.20	90.00	56.25	69.23	97.37	77.08	86.05
airline	85.00	31.48	45.95	95.00	100.0	97.44	91.89	89.47	90.67	100.0	92.11	95.89
airport	100.0	20.00	33.33	100.0	100.0	100.0	100.0	55.56	71.43	100.0	83.33	90.91
capacity	100.0	25.00	40.00	100.0	85.71	92.31	100.0	95.24	97.56	95.24	95.24	95.24
city	00.00	00.00	00.00	100.0	33.33	50.00	100.0	50.00	66.67	100.0	66.67	80.00
distance	100.0	0.100	18.18	100.0	90.00	94.74	100.0	50.00	66.67	100.0	40.00	57.14
flight	91.20	99.10	94.99	99.84	99.21	99.52	100.0	12.50	22.22	94.05	100.0	96.93
flight no	00.00	00.00	00.00	00.00	00.00	00.00	100.0	12.50	22.22	00.00	00.00	00.00
flight time	00.00	00.00	00.00	11.11	100.0	20.00	100.0	100.0	100.0	100.0	100.0	100.0
ground fare	100.0	20.00	33.33	00.00	00.00	00.00	100.0	42.86	60.00	100.0	57.14	72.73
ground service	90.91	94.59	92.72	100.0	100.0	100.0	94.74	100.0	97.30	94.74	100.0	97.30
quantity	100.0	73.68	84.85	25.00	100.0	40.00	25.00	33.33	28.57	100.0	33.33	50.00

From the table5.4, we see that for the SVM model, there are three intent classes *"city"*, *"flight number"*, and *"flight time"*, with 0% precision.

There might be biased classification on SVM predictions for these classes. We also have the feature *"city"* compared to the validation data set.

A similar situation also stands for the deep learning BERT model where intent class *"ground fare"* has 0% precision, indicating biases on this class from the deep learning approach.

We do not have these cases for the RBS and HYBRID SVM-RBS. Giving indications that the RBS approach helps with unbiased predictions.

Overall the performance of models on unseen data is similar to the validation set with top scores on BERT and HYBRID SVM-RBS models.

The table 5.5 below presents the number of true positives(TP), false positives(False Positive (FP)), and false negative(FN) cases on each intent class on validation and test datasets.

Evaluation metrics such as precision, recall, and F1 score are calculated from these parameters.

It is essential to note the large number of false negative cases on rule-based systems compared to other models.
The number of false negatives is more considerable in rule-based systems considering that there are some cases where the rule does not match, especially on test data.

Table 5.5: True positive, False positive and False Negative

	SVM			**BERT**			**RBS**			**Hybrid SVM-RBS**		
	TP	*FP*	*FN*	*TP*	*FP*	*FN*	*TP*	*FP*	*FN*	*TP*	*FP*	*FN*
Validation data												
abbreviation	140	13	7	141	2	6	47	1	11	147	3	0
aircraft	46	7	35	80	2	1	15	1	18	66	3	15
airfare	370	32	53	422	7	1	70	1	68	404	5	19
airline	55	3	102	157	4	0	40	1	27	139	3	18
airport	7	0	13	19	10	1	4	0	5	17	0	3
capacity	9	0	7	15	0	1	5	0	0	16	0	0
city	8	0	11	7	0	12	3	0	0	18	0	1
distance	4	0	16	19	0	1	7	0	0	19	0	1
flight	3640	297	26	3656	7	10	988	25	474	3658	90	8
flight no	0	0	12	10	0	2	3	0	0	6	0	7
flight time	0	0	54	51	3	3	23	3	8	31	0	23
ground fare	5	0	13	15	0	3	2	0	0	14	0	4
ground service	245	21	10	254	3	1	110	4	12	249	5	6
quantity	37	0	14	51	4	0	24	1	4	46	0	5
Test data												
abbreviation	39	9	6	30	0	3	30	1	3	33	0	0
aircraft	9	2	17	8	2	1	2	0	7	4	0	5
airfare	91	12	17	48	1	0	20	3	28	37	1	11
airline	17	3	37	38	2	0	28	1	10	35	0	3
airport	1	0	4	18	2	0	10	0	8	15	0	3
capacity	1	0	3	19	0	2	20	0	1	20	1	1
city	0	0	5	3	0	3	2	0	4	4	0	2
distance	1	0	9	10	0	0	4	0	6	4	0	6
flight	1099	106	10	627	0	5	524	29	10	632	40	0
flight no	0	0	6	8	0	0	0	1	8	0	1	8
flight time	0	0	12	1	2	0	0	2	1	0	4	1
ground fare	1	0	4	5	0	2	0	34	7	1	37	6
ground service	70	7	4	36	2	0	0	4	36	0	1	36
quantity	14	0	5	3	6	0	2	153	1	0	0	3

We are conducting an overlap check between models to reinforce the evaluation insights found so far.
Overlap between two classification models, in our case, means checking if the models predict the same outcome for some instances on the dataset.
This prediction can be a correct prediction or a wrong prediction on the specific intent.
The table 5.6 below presents the ratio between wrongly predicted cases on each model.
We can see the cases in features like "city", "flight no", and "flight time" with the same

number of wrong predictions between models. Giving indications that we can have an overlap between models in these cases.

Table 5.6: Overlap between models. Wrong predicted cases.

	Wong predicted / Overlap check				
	SVM	***RB***	***HYB***	***BERT***	***True label***
Validation data					
abbreviation	3	11	0	2	147
aircraft	27	32	15	1	81
airfare	49	101	19	1	423
airline	68	29	18	0	157
airport	9	5	3	1	20
capacity	5	0	0	3	16
city	**8**	**1**	**1**	**8**	19
distance	13	**1**	**1**	5	20
flight	21	474	8	5	3666
flight no	**12**	**6**	**6**	**12**	12
flight time	54	**23**	**23**	2	54
ground fare	8	11	4	15	18
ground service	8	25	6	2	255
quantity	6	8	5	0	51
Total:	**291**	**727**	**109**	**57**	**4939**
Test data					
abbreviation	0	3	0	0	33
aircraft	6	7	5	1	9
airfare	20	28	11	0	48
airline	16	10	3	0	38
airport	5	8	3	0	18
capacity	**1**	**1**	**1**	3	21
city	**4**	**4**	2	**4**	6
distance	**6**	**6**	**6**	1	10
flight	0	108	0	5	632
flight no	**8**	**8**	**8**	**8**	8
flight time	**1**	0	0	0	1
ground fare	**3**	5	**3**	7	7
ground service	**3**	**3**	0	0	36
quantity	3	**2**	**2**	0	3
Total:	**76**	**193**	**44**	**29**	**870**

Table 5.6 shows that if we consider the total number of wrongly predicted cases, RBS have the most cases, with 727 cases out of 4939 in the validation dataset and 193 out of 870 on the test set.

The BERT model performs best in this case, with 57 out of 4939 in the validation dataset and 29 out of 870 on the test dataset.

Important to note is that there are exceptions in these extremes if we analyze specific intent classes.

In the cases of *"abbreviation"*, *"capacity"*, *"city"*, *"distance"*, *"flight number"*, and *"ground fare"*, the HYBRID SVM-RBS model performed best with the lowest number of wrongly predicted cases on the validation dataset. The table 5.7 below shows the number of correctly predicted cases for each model on the test and validation dataset.

Table 5.7: Overlap between models. Correct predicted cases.

	Correct predicted / Overlap check				
	SVM	***RB***	***HYB***	***BERT***	***True label***
Validation data					
abbreviation	144	136	147	145	147
aircraft	54	49	66	80	81
airfare	374	322	404	422	423
airline	89	128	139	157	157
airport	11	15	17	19	20
capacity	11	**16**	**16**	13	16
city	**11**	**18**	**18**	**11**	19
distance	7	**19**	**19**	15	20
flight	3645	3192	3658	3661	3666
flight no	0	**6**	**6**	0	12
flight time	0	**31**	**31**	52	54
ground fare	10	7	14	3	18
ground service	247	230	249	253	255
quantity	45	43	46	51	51
Total:	**4648**	**4212**	**4830**	**4882**	**4939**
Test data					
abbreviation	**33**	30	**33**	**33**	33
aircraft	3	2	4	8	9
airfare	28	20	37	48	48
airline	22	28	35	38	38
airport	13	10	15	18	18
capacity	**20**	**20**	**20**	18	21
city	**2**	**2**	**4**	**2**	6
distance	**4**	**4**	**4**	9	10
flight	632	524	632	627	632
flight no	**0**	**0**	**0**	**0**	8
flight time	0	1	1	1	1
ground fare	**4**	2	**4**	0	7
ground service	**33**	**33**	**36**	**36**	36
quantity	0	**1**	**1**	3	3
Total:	**794**	**677**	**826**	**841**	**870**

The table 5.7 shows that models predicted quite well in terms of correctly predicting the intents.
We have highlighted in bolt the cases with the exact predictions. This also serves as an indication to check the overlap further.
The test dataset shows that BERT achieved the highest number of correct predictions, with 841 out of 870. SVM and RBS performed similarly, with 794 and 677 correct

predictions, respectively, while HYBRID SVM-RBS achieved the highest number of total predictions (826).

Looking at individual intent classes, BERT achieved the highest accuracy in most classes, except for the *"aircraft"* and *"ground fare"* classes, where and HYBRID SVM-RBS performed better, respectively.

SVM performed poorly in the *"aircraft"*, *"airport"*, and *"city"* classes, while RBS struggled with the *"flight number"* class.

Similarly to the test dataset, the BERT model has the best results on the validation dataset compared to other cases.

Let us look at the individual intents here.

SVM was performing worst with the lowest number of correctly predicted cases.

RBS, which performs quite similarly with HYBRID SVM-RBS on intents like *"capacity"*, *"city"*, *"distance"*, and *"flight number"* has better results than even BERT.

Showing that BERT deep learning model might suffer in predictions for intent classification in these classes.

Now let us go into more detail on our overlap check between the models.

In the table 5.8 below, we have the case of counting cases where only one model predicts correctly, and the rest of the models predict wrong.

The table shows that BERT models progressed in this aspect, considering 94 cases on the validation dataset and 30 cases on the test set.

BERT is the only case where other models predicted correctly and the rest wrong.

Table 5.8: One model predicts correct the rest predict wrong

	One model predicts correct the rest predict wrong				
	SVM	***RB***	***HYB***	***BERT***	***True label***
			Validation data		
abbreviation	0	0	0	0	147
aircraft	0	0	0	15	81
airfare	0	0	0	19	423
airline	0	0	0	18	157
airport	0	0	0	2	20
capacity	0	0	0	0	16
city	0	0	0	0	19
distance	0	0	0	0	20
flight	0	0	0	8	3666
flight no	0	0	0	0	12
flight time	0	0	0	21	54
ground fare	0	0	0	0	18
ground service	0	0	0	5	255
quantity	0	0	0	6	51
Total:	**0**	**0**	**0**	**94**	**4939**
			Test data		
abbreviation	0	0	0	0	33
aircraft	0	0	0	4	9
airfare	0	0	0	11	48
airline	0	0	0	3	38
airport	0	0	0	3	18
capacity	0	0	0	1	21
city	0	0	0	1	6
distance	0	0	0	5	10
flight	0	0	0	0	632
flight no	0	0	0	0	8
flight time	0	0	0	0	1
ground fare	0	0	0	0	7
ground service	0	0	0	0	36
quantity	0	0	0	2	3
Total:	**0**	**0**	**0**	**30**	**870**

Another important case to check for the overlap between models is where two models can predict the same output.

If the models predict the correct intent, this is not a problem.

Nevertheless, if we have cases where models predict the same wrong intent, this is important to be tracked.
Table 5.9 shows the cases where two models predict the same wrong intent.
BERT and HYBRID SVM-RBS models have the lowest number of cases where two models predict the same wrong intent.
Conversely, SVM and HYBRID SVM-RBS have the most cases where both models predict the same wrong intent.

Table 5.9: Two models predict wrong intent but the same prediction

	Two models predict wrong intent but the same prediction				
	RB & SVM	***RB & BERT***	***SVM & BERT***	***SVM & HYB***	***BERT & HYBRID***
Validation data					
abbreviation	0	0	0	0	0
aircraft	6	0	1	15	0
airfare	15	0	0	19	0
airline	12	0	0	18	0
airport	1	1	1	3	1
capacity	0	0	1	0	0
city	1	0	0	1	0
distance	1	0	0	1	0
flight	3	2	0	8	0
flight no	5	1	0	6	0
flight time	16	16	2	23	2
ground fare	3	3	6	4	3
ground service	2	0	0	6	0
quantity	2	0	0	5	0
Total:	**67**	**8**	**11**	**109**	**6**
Test data					
abbreviation	0	0	0	0	0
aircraft	3	0	0	5	0
airfare	10	0	0	11	0
airline	0	0	0	3	0
airport	0	0	0	3	0
capacity	0	0	0	1	0
city	0	0	1	2	0
distance	1	0	0	6	0
flight	0	4	0	0	0
flight no	2	0	0	8	0
flight time	0	0	0	0	0
ground fare	1	3	1	3	1
ground service	0	0	0	0	0
quantity	0	0	0	2	0
Total:	**17**	**7**	**2**	**44**	**1**

Let us see how the RB model performed in comparison with other models.
First, we are focusing on cases where rule-based predict empty cases.
The table 5.10 shows the cases where RB predicted empty and the combination with other models.
From the table, we see that in most cases where RBS predicts empty intent, BERT model will be predicting correct intent.
Also, in the least cases where RBS predicts empty, the SVM and HYBRID SVM-RBS model predicts wrong.

Table 5.10: Rule-based predicts empty intent

	Rule-based predicts empty intent			
	RB empty & SVM wrong	***RB empty & BERT wrong***	***RB empty & SVM correct***	***RB empty & BERT correct***
Validation data				
abbreviation	0	2	11	9
aircraft	8	0	6	14
airfare	3	1	50	52
airline	6	0	10	16
airport	2	0	2	4
capacity	0	0	0	0
city	0	0	0	0
distance	0	0	0	0
flight	5	2	458	461
flight no	0	0	0	0
flight time	7	1	0	6
ground fare	0	0	0	0
ground service	4	0	7	11
quantity	3	0	1	4
Total:	**38**	**6**	**545**	**577**
Test data				
abbreviation	0	0	3	3
aircraft	1	0	2	3
airfare	1	0	13	14
airline	3	0	1	4
airport	3	0	5	8
capacity	1	0	0	1
city	1	2	2	1
distance	5	0	0	5
flight	0	1	104	103
flight no	6	6	0	0
flight time	0	0	0	0
ground fare	1	2	1	0
ground service	0	0	0	0
quantity	2	0	0	2
Total:	**24**	**11**	**131**	**144**

Second, we will track the cases where RB predicts wrong and how this relates to other model's predictions.
The table 5.11 shows similar cases with the case when RBS was predicting empty intents. It is obvious that the number of cases where RBS predicts wrong and SVM wrong is the same as when RBS predicts wrong and HYBRID SVM-RBS predicts wrong.

Table 5.11: Rule-based predicts wrong intent

	Rule-based predicts wrong intent			
	RB wrong & SVM wrong	***RB wrong & BERT wrong***	***RB wrong & SVM correct***	***RB wrong & BERT correct***
Validation data				
abbreviation	0	2	11	9
aircraft	15	0	17	32
airfare	19	1	82	100
airline	18	0	11	29
airport	3	1	2	4
capacity	0	0	0	0
city	1	1	0	0
distance	1	1	0	0
flight	8	4	466	470
flight no	6	6	0	0
flight time	23	2	0	21
ground fare	4	8	7	3
ground service	6	0	19	25
quantity	5	0	3	8
Total:	**109**	**26**	**618**	**701**
Test data				
abbreviation	0	0	3	3
aircraft	5	1	2	6
airfare	11	0	17	28
airline	3	0	7	10
airport	3	0	5	8
capacity	1	0	0	1
city	2	3	2	1
distance	6	1	0	5
flight	0	5	108	103
flight no	8	8	0	0
flight time	0	0	0	0
ground fare	3	5	2	0
ground service	0	0	3	3
quantity	2	0	0	2
Total:	**44**	**23**	**144**	**170**

Third, we present the cases where RBS predicts the correct intent and how this relates to other model's predictions.

Table 5.12 shows that we have the most combination of correct intent classification with other models also predicting the correct intent.

Also, there are a few cases where RBS predicts correctly, and other models predict wrong. It is important to note that there is no case when RBS predicts correctly, and BERT predicts wrong.

Table 5.12: Rule-based predicts correct intent

	Rule-based predicts correct intent			
	RB correct & SVM wrong	***RB correct & BERT wrong***	***RB correct & SVM correct***	***RB correct & BERT correct***
	Validation data			
abbreviation	3	0	133	136
aircraft	12	1	37	48
airfare	30	0	292	322
airline	50	0	78	128
airport	6	0	9	15
capacity	5	3	11	13
city	7	7	11	11
distance	12	4	7	15
flight	13	1	3179	3191
flight no	6	6	0	0
flight time	31	0	0	31
ground fare	4	7	3	0
ground service	2	2	228	228
quantity	1	0	42	43
Total:	**182**	**31**	**4030**	**4181**
	Test data			
abbreviation	0	0	30	30
aircraft	1	0	1	2
airfare	9	0	11	20
airline	13	0	15	28
airport	2	0	8	10
capacity	0	3	20	17
city	2	1	0	1
distance	0	0	4	4
flight	0	0	524	524
flight no	0	0	0	0
flight time	1	0	0	1
ground fare	0	2	2	0
ground service	3	0	30	33
quantity	1	0	0	1
Total:	**32**	**6**	**645**	**671**

In the table below, we will compare SVM, BERT, and HYBRID SVM-RBS models.
The table shows every combination of these models when they predict correct and wrong between each other.
The table shows that we have the biggest number of predictions with a slight difference in cases where SVM predicts correct and BERT predicts correct but also when SVM predicts correct, and HYBRID SVM-RBS predicts correct intent.
However, the lowest number of cases when SVM predicts wrong and BERT wrong intents.
It is important to note that there is no case when SVM predicts correctly and HYBRID SVM-RBS predicts wrong.
Also, there are some cases when SVM predicts correctly, and BERT predicts wrong.
Even though the general evaluation of the model BERT has far greater results than the

SVM.

Table 5.13: SVM comparison to BERT and HYBRID

	SVM comparison to BERT and HYBRID							
	SVM *wrong &* ***BERT*** *correct*	***SVM*** *wrong &* ***BERT*** *wrong*	***SVM*** *wrong &* ***HYBRID*** *correct*	***SVM*** *wrong &* ***HYBRID*** *wrong*	***SVM*** *correct &* ***BERT*** *correct*	***SVM*** *correct &* ***BERT*** *wrong*	***SVM*** *correct &* ***HYBRID*** *correct*	***SVM*** *correct &* ***HYBRID*** *wrong*
Validation data								
abbreviation	3	0	3	0	142	2	144	0
aircraft	26	1	12	15	54	0	54	0
airfare	49	0	30	19	373	1	374	0
airline	68	0	50	18	89	0	89	0
airport	8	1	6	3	11	0	11	0
capacity	4	1	5	0	9	2	11	0
city	5	3	7	1	6	5	11	0
distance	8	5	12	1	7	0	7	0
flight	21	0	13	8	3640	5	3645	0
flight no	0	12	6	6	0	0	0	0
flight time	52	2	31	23	0	0	0	0
ground fare	0	8	4	4	3	7	10	0
ground service	8	0	2	6	245	2	247	0
quantity	6	0	1	5	45	0	45	0
Total:	***258***	***33***	***182***	***109***	***4624***	***24***	***4648***	***0***
Test data								
abbreviation	0	0	0	0	33	0	33	0
aircraft	5	1	1	5	3	0	3	0
airfare	20	0	9	11	28	0	28	0
airline	16	0	13	3	22	0	22	0
airport	5	0	2	3	13	0	13	0
capacity	1	0	0	1	17	3	20	0
city	2	2	2	2	0	2	2	0
distance	5	1	0	6	4	0	4	0
flight	0	0	0	0	627	5	632	0
flight no	0	8	0	8	0	0	0	0
flight time	1	0	1	0	0	0	0	0
ground fare	0	3	0	3	0	4	4	0
ground service	3	0	3	0	33	0	33	0
quantity	3	0	1	2	0	0	0	0
Total:	***61***	***15***	***32***	***44***	***780***	***14***	***794***	***0***

In the table below, we will compare BERT and HYBRID SVM-RBS models.
The table shows every combination of these models when they predict correct and wrong between each other.
The table shows that in most cases, BERT and HYBRID SVM-RBS predict correctly on intent classification.
It is important to note that there are very few cases where both models can predict wrong intents.
These cases are seen in the intent classes such as *"airport"*, *"city"*, *"distance"*, *"flight number"*, *"flight time"*, and *"ground fare"* on cases of validation dataset.
In the test dataset, the cases where both models predict wrong are seen in *"aircraft"*, *"city"*, *"distance"*, and *"ground fare"* classes.

Table 5.14: BERT comparison to HYBRID

	BERT comparison to HYBRID			
	BERT correct & HYB correct	***BERT correct & HYB wrong***	***BERT wrong & HYB correct***	***BERT wrong & HYB wrong***
Validation data				
abbreviation	145	0	2	0
aircraft	65	15	1	0
airfare	403	19	1	0
airline	139	18	0	0
airport	17	2	0	1
capacity	13	0	3	0
city	11	0	7	1
distance	15	0	4	1
flight	3653	8	8	0
flight no	0	0	6	6
flight time	31	21	0	2
ground fare	3	0	11	4
ground service	247	6	2	0
quantity	46	5	0	0
Total:	***4788***	***94***	***42***	***15***
Test data				
abbreviation	33	0	0	0
aircraft	4	4	0	1
airfare	37	11	0	0
airline	35	3	0	0
airport	15	3	0	0
capacity	17	1	3	0
city	1	1	3	1
distance	4	5	0	1
flight	627	0	5	0
flight no	0	0	0	8
flight time	1	0	0	0
ground fare	0	0	4	3
ground service	36	0	0	0
quantity	1	2	0	0
Total:	***811***	***30***	***15***	***14***

From the quantitative evaluation, we saw that we are dealing with an unbalanced dataset.
We saw many cases where the same word tag was used on the dataset.
The frequency of words was essential in defining the rules.
The dataset that we are working on contains three parts train dataset, validation dataset, and unseen data for the models on the test dataset.
The number of true positives, true negatives, and false negatives plays an essential role in calculating evaluation metrics such as accuracy, precision, recall, and F1 score.
Since we are dealing with an unbalanced dataset, we need to consider the impact of the dominating class, in our case, the intent *"flight"*.
We use macro, and weighted averaging to weight the intent classes.
During the quantitative evaluation, we also did an overlap check between models, checking

every combination between models and their comparisons. As a result, some cases could be biased on the SVM model and overlap cases on specific intents on the models. Interesting to point out is that besides that BERT and HYBRID SVM-RBS had the best accuracy percentage on intent predictions, there were also some cases where HYBRID SVM-RBS was classifying the correct intent and BERT wrong intent.

5.2 Qualitative evaluation

This section will present the qualitative analysis of our intent classification task.
With qualitative analysis, we intend to identify patterns that lower the error rate by correctly classifying the wrongly predicted intents.
In the qualitative analysis, we will analyze the classification of our models on concrete examples to see the possibility of identifying the reason why these cases are wrongly predicted.
The table 5.15 below shows the cases of wrongly predicted intents from RBS models.
We can see from the table 5.15 the sentences with the wrong classification from the RBS model.
By analyzing the sentences, we see that some of the sentences with word tags below, to other intents, tend to be wrongly predicted.
To lower this error rate for the RBS model, we can exclude the particular word tags from the rules defined on the RB model.

Table 5.15: Wrongly predicted cases on RB example

Sentence[MS CNTK19]	RB	SVM	BERT	Intent
on flight us air 2153 from san francisco to baltimore what time and what city does the plane stop in between	flight_time	flight	flight	flight
on usa air how many flights leaving oakland on july twenty seventh to boston nonstop	quantity	flight	quantity	flight
i ’d like to travel from boston to baltimore on us air 269 please tell me the times	flight_time	flight	flight	flight
find me the earliest boston departure and the latest atlanta return trip so that i can be on the ground the maximum amount of time in atlanta and return to boston on the same day	ground_service	flight	flight	flight
show me the lowest price from dallas to baltimore	airfare	airfare	flight	flight
show me the daily flight schedule between boston and pittsburgh	flight_time	flight	flight_time	flight
how can i get from boston to atlanta and back in the same day and have the most hours on the ground in atlanta	ground_service	flight	flight	flight
eastern flies from atlanta to denver what type of aircraft do you use before 6 pm	aircraft	aircraft	flight	flight
on united airlines flying from denver to san francisco before 10 am what type of aircraft is used	aircraft	aircraft	flight	flight
what classes of service does twa have	abbreviation	flight	flight	flight
what classes of service does twa provide	abbreviation	flight	flight	flight

The table 5.16 shows the correctly predicted cases from all the models.
It is important to show the correctly predicted cases. The difficulty stands on the part of the "black box" for the BERT model.
However, as seen from the example below, the RBS and HYBRID SVM-RBS models can develop similar outcomes as BERT.
The difference is that tracking the error rate and improving the model is easier than BERT.

Table 5.16: Correctly predicted examples

Sentence[MS CNTK19]	RB	HYBRID	SVM	BERT	Intent
what kinds of planes are used by american airlines	aircraft	aircraft	aircraft	aircraft	aircraft
what types of aircraft does delta fly	aircraft	aircraft	aircraft	aircraft	aircraft
is there a plane from boston to washington	aircraft	aircraft	flight	aircraft	aircraft
what ’s the smallest plane that flies from pittsburgh to baltimore on eight sixteen	aircraft	aircraft	flight	aircraft	aircraft
repeating leaving denver to san francisco before 10 am what type of aircraft is used	aircraft	aircraft	aircraft	aircraft	aircraft
what type of aircraft flies from pittsburgh to baltimore	aircraft	aircraft	aircraft	aircraft	aircraft
what type of plane is an m80	aircraft	aircraft	aircraft	aircraft	aircraft
show me the type of aircraft that cp uses	aircraft	aircraft	aircraft	aircraft	aircraft
what type of aircraft does eastern fly from atlanta to denver before 6 pm	aircraft	aircraft	aircraft	aircraft	aircraft
what type of aircraft leaving after 2 pm from boston to oakland	aircraft	aircraft	aircraft	aircraft	aircraft
kindly give me the type of aircraft used to fly from atlanta to denver	aircraft	aircraft	aircraft	aircraft	aircraft
what kind of aircraft does delta fly before 8 am on august second from boston to denver	aircraft	aircraft	aircraft	aircraft	aircraft
what type of aircraft is used on american airline flight 315	aircraft	aircraft	aircraft	aircraft	aircraft
what is the type of aircraft for united flight 21	aircraft	aircraft	aircraft	aircraft	aircraft

The table 5.16 shows the correctly predicted cases from all the models.
It is important to show the correctly predicted cases. The difficulty stands on the part of the "black box" for the BERT model.
However, the example above 5.16 shows that the RBS and HYBRID SVM-RBS models can develop similar outcomes as BERT.

The difference is that tracking the error rate and improving the model is easier than BERT.

Table 5.17: One model predicts correctly the rest of the model wrong

Sentence[MS CNTK19]	RB	HYBRID	SVM	BERT	Intent
what is the arrival time in san francisco for the 755 am flight leaving washington	flight	flight	flight	flight_time	flight_time
show me times for flights from san francisco to atlanta		flight	flight	flight_time	flight_time
i would like the time of all flights from san francisco to pittsburgh on sunday	flight	flight	flight	flight_time	flight_time
please tell me the times of the flights between boston and baltimore	flight	flight	flight	flight_time	flight_time
show me times for coach flights between boston and baltimore on wednesday		flight	flight	flight_time	flight_time
what is the departure time of the latest flight of united airlines from denver to boston	flight	flight	flight	flight_time	flight_time
now i 'd like a schedule for the flights on tuesday morning from oakland no from dallas fort worth to atlanta	flight	flight	flight	flight_time	flight_time
what time does the tuesday morning 755 flight leaving washington arrive in san francisco	flight	flight	flight	flight_time	flight_time
what time are the flights leaving from denver to pittsburgh on july seventh	flight	flight	flight	flight_time	flight_time
when does continental fly from philadelphia to denver on sundays		flight	flight	flight	flight_time
what time are the flights from baltimore to san francisco	flight	flight	flight	flight_time	flight_time
what time does the flight leave denver going to san francisco on continental airlines	flight	flight	flight	flight_time	flight_time
what is delta 's schedule of morning flights to atlanta		flight	flight	flight_time	flight_time
what is american 's schedule of morning flights to atlanta		flight	flight	flight_time	flight_time

The table 5.17 below shows cases where only BERT predicts the correct intent.
We see that some cases predict the exact wrong prediction, meaning that SVM and

HYBRID SVM-RBS models predict the same wrong case.
The same wrong prediction on the SVM model led to the first bias cases.
The same cannot be concluded for RB since we also have empty cases on the RBS model.
The table shows that the SVM model's intent class "flight time" is biased with the "flight" class.
Due to the similarity of sentences with flight time intent and flight intent, SVM cannot differentiate between these cases.
We will also see in the table 5.18 below that on the feature "flight no", besides SVM, we also have the BERT model, which is biased on this feature with "flight_time" or "quantity" intent class. Compared to RBS, which sometimes produces wrong predictions but does not show biases.

Table 5.18: Models predict the same wrong intent

Sentence[MS CNTK19]	RB	HYBRID	SVM	BERT	Intent
flight numbers from columbus to minneapolis tomorrow	flight_no	flight_no	flight	flight_time	flight_no
i 'm trying to find the flight number from a flight from orlando to cleveland on us air and it arrives around 10 pm	flight	flight	flight	flight_time	flight_no
flight numbers from minneapolis to long beach on june twenty six	flight_no	flight_no	flight	flight_time	flight_no
please show me the return flight number from toronto to st. petersburg	flight	flight	flight	flight_time	flight_no
what is the flight number for the continental flight which leaves denver at 1220 pm and goes to san francisco	flight	flight	flight	flight_time	flight_no
what is the number of first class flights on american airlines	flight_no	flight_no	flight	quantity	flight_no
may i have a listing of flight numbers from columbus ohio to minneapolis minnesota on monday	flight_no	flight_no	flight	flight_time	flight_no
which is the flight number for the us air flight from philadelphia to boston is it 279 or is it 137338	flight_no	flight_no	flight	flight_time	flight_no
what is the flight number of the earliest flight between boston and washington dc	flight	flight	flight	flight_time	flight_no
what are the flight numbers of the flights which go from san francisco to washington via indianapolis	flight	flight	flight	flight_time	flight_no

The table 5.19 below shows the cases where we have empty intent classification from the RBS model.
Analyzing the sentences, we see some cases that are not part of any rules defined in the RBS model.

A workaround for these cases is to develop rules that include these cases to classify the intent properly.

Table 5.19: RB predicts empty examples

Sentence[MS CNTK19]	RB	HYBRID	SVM	BERT	Intent
code ff		abbreviation	abbreviation	abbreviation	abbreviation
i would like a list of flights from pittsburgh to dallas		flight	flight	flight	flight
on november twenty third of this year 1991 i 'd like to fly from atlanta to denver and i 'd like to fly on delta		flight	flight	flight	flight
do you have any airlines that would stop at denver on the way from baltimore to san francisco		flight	flight	airline	airline
give me flights from san francisco to boston on thursday afternoon		flight	flight	flight	flight
denver to atlanta		flight	flight	flight	flight
i would like information on flights leaving atlanta in the afternoon arriving in dallas		flight	flight	flight	flight
dallas to baltimore		flight	flight	flight	flight
may i have a listing of flights on monday from minneapolis to long beach california please		flight	flight	flight	flight
show me times for coach flights between boston and baltimore on wednesday		flight	flight	flight_time	flight_time
what is airline dl		airline	airline	airline	airline
does delta airlines fly from boston to washington dc		flight	flight	flight	flight
do you fly a 747 from baltimore to san francisco		flight	flight	flight	flight

The tables 5.20 5.21 below show the cases of RB predicting a correct and respectfully wrong intent classification.
Giving us indications that human expertise plays a crucial role in defining the rules.
Human expertise is responsible for modeling the rule-based system to be cautious that the number of empty predicted, respectively wrong predicted cases is as low as possible.

Table 5.20: RB predicts wrong examples

Sentence[MS CNTK19]	RB	HYBRID	SVM	BERT	Intent
please list all airline flights between denver and boston	flight	flight	flight	airline	airline
does any airline have an afternoon flight from boston to oakland	flight	flight	flight	airline	airline
what kind of airline is flight ua 281 from boston to denver	flight	flight	flight	airline	airline
what airline is the flight originating in atlanta on november seventh at noon and arriving in san francisco at 210 pm	flight	flight	flight	airline	airline
what does the airline code dl stand for	abbreviation	abbreviation	abbreviation	airline	airline
what kind of airline is flight ua 281 from boston to denver	flight	flight	flight	airline	airline
which airline has the most business class flights	flight	flight	airline	airline	airline
does any airline have an early afternoon flight from boston to pittsburgh	flight	flight	flight	airline	airline
what airlines fly from st. petersburg to milwaukee and from milwaukee to tacoma	flight	flight	flight	airline	airline
does any airline have an early afternoon flight from boston to denver	flight	flight	flight	airline	airline
does any airline have a jet flight between pittsburgh and baltimore	flight	flight	flight	airline	airline
is there an airline that has a flight from philadelphia to san francisco with a stop in dallas	flight	flight	flight	airline	airline
does any airline have an afternoon flight from atlanta to boston	flight	flight	flight	airline	airline

Table 5.21: RB predicts correct examples

Sentence[MS CNTK19]	RB	HYBRID	SVM	BERT	Intent
what is fare code h	abbreviation	abbreviation	abbreviation	abbreviation	abbreviation
what is booking class c	abbreviation	abbreviation	abbreviation	abbreviation	abbreviation
what does fare code q mean	abbreviation	abbreviation	abbreviation	abbreviation	abbreviation
what is fare code qw	abbreviation	abbreviation	abbreviation	abbreviation	abbreviation
what does the fare code f mean	abbreviation	abbreviation	abbreviation	abbreviation	abbreviation
what is fare code h	abbreviation	abbreviation	abbreviation	abbreviation	abbreviation
what does fare code qw mean	abbreviation	abbreviation	abbreviation	abbreviation	abbreviation
what does mco stand for	abbreviation	abbreviation	abbreviation	abbreviation	abbreviation
what 's the difference between fare code q and fare code f	abbreviation	abbreviation	abbreviation	abbreviation	abbreviation
what is the yn code	abbreviation	abbreviation	abbreviation	abbreviation	abbreviation
what is ord	abbreviation	abbreviation	abbreviation	abbreviation	abbreviation
what 's fare code yn	abbreviation	abbreviation	abbreviation	abbreviation	abbreviation
what does restriction ap 57 mean	abbreviation	abbreviation	abbreviation	abbreviation	abbreviation
what is bna	abbreviation	abbreviation	abbreviation	abbreviation	abbreviation
explain the restriction ap 80	abbreviation	abbreviation	abbreviation	abbreviation	abbreviation
what does the abbreviation dl mean	abbreviation	abbreviation	abbreviation	abbreviation	abbreviation
what is the fare code y and what is the fare code h	abbreviation	abbreviation	abbreviation	abbreviation	abbreviation
what does ua stand for	abbreviation	abbreviation	abbreviation	abbreviation	abbreviation
what is fare code m	abbreviation	abbreviation	abbreviation	abbreviation	abbreviation
what is sa	abbreviation	abbreviation	abbreviation	abbreviation	abbreviation

We mentioned in the quantitative analysis that there are some cases where the HYBRID SVM-RBS model correctly classifies the intent class compared to the BERT model, which in terms of evaluation metrics, is performing slightly better than the HYBRID SVM-RBS model.
The tables 5.23 5.22 below show examples where HYBRID SVM-RBS is performing better than BERT and the other way around.

Table 5.22: HYBRID predicts correct intent and BERT wrong examples

Sentence[MS CNTK	19]RB	HYBRID	SVM	BERT	Intent
define airline us		abbreviation	abbreviation	airline	abbreviation
define airline ua		abbreviation	abbreviation	airline	abbreviation
i 'm going to leave philadelphia and i want to go to san francisco and i want to fly first class american and i want a stop in dallas can you please tell me what type of aircraft you will be flying	aircraft	aircraft	flight	flight	aircraft
what do these cost		airfare	airfare	abbreviation	airfare
list number of people that can be carried on each type of plane that flies between pittsburgh and baltimore	capacity	capacity	aircraft	aircraft	capacity
how many people fly on a turboprop	capacity	capacity	capacity	quantity	capacity
how many passengers can a boeing 737 hold	capacity	capacity	capacity	quantity	capacity
where is mco	city	city	city	airline	city
where is general mitchell international located	city	city	city	airline	city
where is general mitchell international located	city	city	city	airline	city
where is lester pearson airport	city	city	city	airport	city
is bwi washington	city	city	city	airline	city
what time zone is denver in	city	city	ground_service	flight_time	city
are there any other cities that i can fly from boston to dallas through that i can get a flight earlier than 1017 in the morning	flight	flight	flight	airline	city

Table 5.23: Bert predicts correct and Hybrid wrong example

Sentence[MS CNTK19]	RB	HYBRID	SVM	BERT	Intent
what is the cost of a round trip flight from pittsburgh to atlanta beginning on april twenty fifth and returning on may sixth	flight	flight	flight	airfare	airfare
i would like the least expensive airfare flight on sunday to pittsburgh from san francisco	flight	flight	flight	airfare	airfare
i would like your rates between atlanta and boston on september third		flight	flight	airfare	airfare
please list fares for all the flights from atlanta to philadelphia on august the first	flight	flight	flight	airfare	airfare
show prices for all flights from baltimore to dallas on july twenty ninth	flight	flight	flight	airfare	airfare
i need a first class ticket on united airlines from denver to baltimore scheduled for december seventeenth	flight	flight	flight	airfare	airfare
i 'd like information on the least expensive airfare round trip from pittsburgh to boston		flight	flight	airfare	airfare
how much is the 718 am flight from las vegas to new york twa	flight	flight	flight	airfare	airfare
please list the cost of all flights from philadelphia to denver airport next sunday	flight	flight	flight	airfare	airfare
what is the fare on the first flight from atlanta to denver on thursday morning	flight	flight	flight	airfare	airfare
can you show me the price of a flight to washington from atlanta on thursday morning	flight	flight	flight	airfare	airfare
how much is the cheapest flight from denver to pittsburgh with a stop in atlanta	flight	flight	flight	airfare	airfare
let 's see how much would a direct flight from atlanta to denver be on may seventh	flight	flight	flight	airfare	airfare
what are the prices of the flights from atlanta to dallas in the morning	flight	flight	flight	airfare	airfare
display all fare codes	flight	flight	abbreviation	airfare	airfare
please give me the prices for all flights from philadelphia to denver airport next sunday	flight	flight	flight	airfare	airfare
what united airlines first class airfare flights are available from denver to baltimore on july three	flight	flight	flight	airfare	airfare

The tables show many cases where comparing BERT and HYBRID SVM-RBS, depending on the specific intent class, one performs better than the other.
Considering this case, one approach to lower the error on the HYBRID SVM-RBS model is to generate more rules to lower the number of wrongly predicted intents.
This explainability of the HYBRID SVM-RBS model gives advantages compared to the BERT model's "black box" effect on intent classification. Different examples on different intent class are presented in Appendix A.2 for further qualitative analysis.

5.3 Contribution to the state-of-the-art

The table shows some state-of-the-art approaches regarding intent classification on the ATIS dataset.
We can see our deep learning models using BERT and our HYBRID SVM-RBS. Produced leading results compared to these other approaches.
Important to note is our contribution to biased predictions, explainability, and interpretability of results compared to other approaches on the benchmark.

Table 5.24: Contribution to the state-of-the-art

	ATIS
Model	**Intent Accuracy**
Joint Seq [HTTC+16]	92.6
Attention BiRNN [LL16]	91.1
Slot-Gated Full Atten [GGH+18]	93.6
Slot-Gated Intent Atten [GGH+18]	94.1
Self-Attentive Model [LLQ18]	96.8
Bi-Model [WSJ18]	96.4
CAPSULE-NLU [ZLD+19]	95.0
SF-ID Network [ENCS19]	96.6
Our BERT	99.14
Our HYBRID SVM-RBS	97.79

CHAPTER 6

Conclusion

In this master's thesis, we proposed a novel HYBRID SVM-RBS intent classification system built on a classic machine learning SVM and a RBS combination.

Rule-based system rules are defined using a syntactic graph representation of text.
Using a syntactic graph representation of text, we can see the syntactic relation between word tags in a sentence.

Initially, we presented a rule-based approach where we defined rules using POTATO explainable artificial framework.
From POTATO, we observed that although the initial results were good.
We had a considerable number of false negatives cases assigned.
By editing the rules manually, we were able to initially decrease the number of false negative cases and this way to increase recall.

For comparison, we used a deep learning pre-trained BERT model on intent classification.
The BERT model performs best on the benchmark.
However, the BERT has a disadvantage in maintaining the model, considering that besides the parameters on training the model, the training process is a "black box" process that is very complex and difficult to edit and maintain.
It can also be computationally expensive since it takes considerable time to train the model.

Another approach for comparison that we used is the machine learning SVM approach.
As a classic classification approach in our case, SVM performed better than the rule-based system alone.
It had the advantage of a short and simple training process for the model and better performance than a rule-based system. Nevertheless, more is needed compared to the

deep learning pre-trained BERT model performance.

Considering the advantages and disadvantages of rules-based and SVM and the benefits of defining the rules from a syntactic graphical text representation. We defined a HYBRID SVM-RBS model from SVM and a rule-based model.
The HYBRID SVM-RBS model performed very well on the benchmark, very comparable with the deep learning BERT model.
The advantages of the HYBRID SVM-RBS model compared to BERT were that it is more straightforward to maintain the model, and we can present the model from performing any hidden biases.

Our evaluation of the models is based on quantitative and qualitative analysis.

From the quantitative evaluation, we conclude that since we are working with an unbalanced dataset, it is crucial to consider averaging since the impact of unbalanced features affects the evaluation metrics.
We concluded that the frequency of word distribution is an essential step in constructing the rules since it contributes to generalizing them.
We also saw many cases where models predicted the same wrong classification of intents. The wrong prediction led to the understanding that we can face potential hidden biases in both machine and deep learning approaches.
The HYBRID SVM-RBS model was not facing any hidden biases generated from RBS, and this is due to human interaction impact on the model. However, since the SVM was facing biased predictions, it will also affect the cases predicted from SVM on HYBRID SVM-RBS model. Giving us indications that also HYBRID SVM-RBS can face biased predictions.
Since rules are defined by human expertise, analyzing the data with a syntactic graph representation was performed to avoid any biased rule prediction before defining them.

Qualitative analysis showed us concrete patterns which indicate the error rate on model classification. Furthermore, this error rate indicates on decreasing the performance of the model.
It's important to note that we also observed instances in which the machine learning SVM model contained undetected biases.
Qualitative analysis also impacted proving the insights we observed during the quantitative evaluation.

We conclude with the following answers to the research questions during our work.

1. *How does a rule-based syntactic graph system perform on the intent classification task?*

The rule-based system using syntactic graphs has the benefit of producing unbiased results. On the benchmark, in terms of precision, it performs in specific class intent cases even better than machine learning or deep learning approaches.
Rule-based systems have the advantage of their performance since they use predefined rules, but there is always the risk of missing the predictions on unseen data.

2. *How do graph-based methods compare to simple ML baselines?*
   Graph-based methods can be comparable in evaluation results with ML baselines. It is a significant advantage that rule-based systems do not suffer from hidden biased predictions.
   The disadvantage compared to ML baselines is that the rule-generation process requires expertise in the field and can be complex.

3. *What are the bottlenecks of rule-based systems, and what syntactic patterns characterize the main error classes?*
   The bottleneck of rule-based systems with syntactic graph representation is the generalization of rules.
   Since creating the rules is complex, the idea is to generalize the rules as much as possible to consider as many combinations as possible from unseen data.
   During our experiments, we faced many cases where rules did not predict any case on unseen data. Therefore, unpredicted intent in such cases is due to the unknown effect of the unseen data on the model, and we consider it as a factor that indicates the increase in error rate.

APPENDIX A

Appendix

A.1 Rule-Based System Intent Classification Rules

Table A.1: Rules used on Rule-based system

Rules	Intent
[['(u_55/show:obj(trip|itinerary|flight|departure))'], [], "flight"],	Flight
[['(u_2/root:root(u_55/show:obj(trip|itinerary|flight|departure)))'], [], "flight"],	Flight
[['(u_55/show:iobj(we|I):obj(trip|itinerary|flight))'], [], "flight"],	Flight
[['(u_18/flight:det(that|which|all|any|the|milwaukee|what|a))'], [], "flight"],	Flight
[['(u_2/root:root(request|meal|X10|florida|atlanta|interested|chicago|start|miami|want|wish|louis|return|connect|sfo|information|live|newark|need|thank|wednesday|how|seattle|flight|arrive| petersburg|make|we|to|sorry|when|charlotte|listing|carry|toronto|vegas| display|philadelphia))'], [], "flight"],	Flight
[['(u_2/root:root(u_85/list:obj(landing|takeoff|all|trip|flight)))'], [], "flight"],	Flight
[['(u_3/root:root (u_50/show:obj(airfare|ticket|cost|fare|price)))'], [], 'airfare'],	Airfare
[['(u_18/flight:case(for|of|on):nmod-of(cost|fare|price|airfare))'], [], 'airfare'],	Airfare
[['(u_159/cost:xcomp(go|fly|travel):aux(u_71/do))'], [], 'airfare'],	Airfare
[['(u_135/ticket:det(what|the|a))'], [], 'airfare'],	Airfare
[['(u_15/root:root(much|airfare|ticket|fare|price))'], [], 'airfare'],	Airfare
[['(u_15/root:root(u_49/show:obj(airfare|ticket|cost|fare|price)))'], [], 'airfare'],	Airfare
[['(u_49/show:obj(fare|cost|ticket|price))'], [], 'airfare'],	Airfare
[['(u_12/fare:compound(thrift|cost|air|trip))'], [], 'airfare'],	Airfare
[['(u_14/fare:det(u_13/the):nsubj-of(u_41/what))'], [], 'airfare'],	Airfare
[['(u_41/what:nsubj(fare|ticket|price|airfare):cop(u_32/be))'], [], 'airfare'],	Airfare
[['(u_3/root:root(u_41/what:nsubj(fare|ticket|price|airfare)))'], [], 'airfare'],	Airfare
[['(u_50/show:iobj(we|I):obj(fare|cost|ticket|price))'], [], 'airfare'],	Airfare
[['(u_159/cost:xcomp(go|fly|travel):aux(u_71/do))'], [], 'airfare'],	Airfare
[['(u_5/fly:mark(u_4/to):xcomp-of(u_159/cost))'], [], 'airfare'],	Airfare
[['(u_3/root:root(u_159/cost:xcomp(go|take|fly|travel)))'], [], 'airfare'],	Airfare
[['(u_3/root:root(much|airfare|ticket|fare|price))'], [], 'airfare'],	Airfare
[['(u_80/price:nmod(ea|we|morning|economy|ticket|class|air|seat|flight|fare))'], [], 'airfare'],	Airfare
[['(u_41/what:nsubj(fare|ticket|price|airfare))'], [], 'airfare'],	Airfare
[['(u_12/fare:det(u_13/the):nsubj-of(u_41/what))'], [], 'airfare'],	Airfare
[['(u_117 / ground)'], [], "ground_service"],	Ground Service
[['(u_116 / transportation)'], [], "ground_service"],	Ground Service
[['(u_220 / car)'], [], "ground_service"],	Ground Service
[['(u_116 / transportation :compound (u_117 / ground))'], [], "ground_service"],	Ground Service
[['(u_118 / airport :case (u_125 / at))'], [], "ground_service"],	Ground Service
[['(u_321 / transport)'], [], "ground_service"]]	Ground Service
[['(u_29 / airline :det (which|the|what))'], [], "airline"],	Airline
[['(u_15 / root :root (u_49 / show :obj (airline|abbreviation|name)))'], [], "airline"],	Airline
[['(u_49 / show :iobj (u_1 / I) :obj (u_29 / airline))'], [], "airline"],	Airline
[['(u_3 / fly :nsubj (u_29 / airline :det (u_41 / what)))'], [], "airline"],	Airline
[['(u_49 / show :obj (airline|abbreviation|name))'], [], "airline"],	Airline
[['(u_3 / airline)'], ["(u_3 / we|leave|do|ua|stop|be|and)","(u_4 / flight)","(u_8 / to|Canadian)","(u_1 / what)","(u_6 / Canadian)"], 'airline']	Airline
[['(u_172 / code :compound (yn|fare|meal))'], [], "abbreviation"],	Abbreviation
[['(u_311 / mean :obj (u_41 / what))'], [], "abbreviation"],	Abbreviation
[['(u_15 / root :root (restriction|explain|mean))'], [], "abbreviation"],	Abbreviation
[['(u_15 / root :root (u_311 / mean :obj (u_41 / what)))'], [], "abbreviation"],	Abbreviation
[['(u_222 / stand :aux (u_72 / do))'], [], "abbreviation"],	Abbreviation
[["(u_1 / what)"], ["(u_4 / aircraft|plane|ap57|know|in|unite|use|least|cost|a|in|airplane|car|at|continental|ap68|boston|co|meal|mco|ap80)", "(u_5 / from)","(u_7 / to)","(u_9 / flight)","(u_3 / the)","(u_2 / ground|airline)"], "abbreviation"],	Abbreviation
[['(u_15 / root :root (u_41 / what :nsubj (m|d10|hp|ewr|meaning|ord|abbreviation|bur|ap|y|difference|code)))'], [], 'abbreviation']	Abbreviation
[['(u_17/root:root(co|m80|type|inform))'], [], "aircraft"],	Aircraft
[['(u_17/root:root(u_116/use:auxCOLONpass(u_44/be)))'], [], "aircraft"],	Aircraft
[['(u_148/aircraft:case(u_68/of):nmod-of(u_211/type|kind))'], ["(u_2 / united|eastern)","(u_1 / eastern)"], "aircraft"],	Aircraft
[['(u_49 / kind :det (u_17 / what))'], ["(u_5 / transportation)","(u_4 / airline)"], 'aircraft'],	Aircraft
[['(u_2 / be :nsubj (plane|aircraft))'], [], 'aircraft'],	Aircraft
[['(u_1 / what :cop (u_2 / be) :nsubj (.* / aircraft|type|plane))'], [], 'aircraft'],	Aircraft
[['(u_211/type:nmod(airplane|airline|capacity|aircraft))'], ["(.* / flight)"], "aircraft"]	Aircraft
[['(u_3 / time :det (a|the|what))'], ["(.* / fly|cheapest|same|transportation)"], "flight_time"],	Flight Time
[['(u_301 / schedule :det (.*))'], ['(u_7 / transportation)'], 'flight_time']	Flight Time
[['(u_2 / many :mark (u_1 / how) :amod-of(city|flight|airline|we|stop))'], [], "quantity"],	Quantity
[['(u_2 / many :advmod (u_1 / how) :amod-of(airport|flight|class|code|fare|we|stop))'], [], "quantity"]	Quantity

[['(u_0 / root :root (.* / airport))'], ["(.* / what|be|the)"], "airport"],	Airport
[['(u_2 / airport :det (u_1 / what))'], ["(.* / airline)"], "airport"],	Airport
[['(u_1 / give|show :iobj (u_2 / I) :obj (.* / list|airport))'], ["(.* / airline|rental)","(.* / transportation)","(u_6 / flight)"], "airport"],	Airport
[['(u_1 / what :cop (u_2 / be) :nsubj (u_4 / airport|name))'], [], 'airport']	Airport
[['(u_1 / tell :iobj (u_2 / I) :obj (u_3 / distance) :obl (u_6 / airport :case (u_4 / from) :compound (u_5 / orlando)))'], [], "distance"],	Distance
[['(u_2 / far|long :advmod (u_1 / how|paul))'], ["(u_6 / transportation)"], "distance"],	Distance
[['(u_1 / what :cop (u_2 / be) :nsubj (u_4 / distance))'], [], "distance"]	Distance
[['(u_3 / washington :cop (u_1 / be) :compound (u_2 / bwi) :root-of (u_0 / root)))'], [], "city"],	City
[['(u_3 / zone :det (u_1 / what) :compound (u_2 / time))'], [], "city"],	City
[['(u_1 / what :cop (u_2 / be) :nsubj (u_4 / city :det (u_3 / the)))'], [], "city"],	City
[['(u_1 / show :iobj (u_2 / I) :obj (u_4 / city))'], [], "city"],	City
[['(u_2 / city :det (u_1 / what|which))'], [], "city"],	City
[['(u_1 / where :cop (u_2 / be))'], [], "city"],	City
[['(u_1 / be :expl (u_2 / there) :nsubj (u_5 / city :det (u_3 / any)))'], [], "city"]	City
[['(u_2 / much :advmod (u_1 / how) :advmod-of (u_5 / cost))'], ["(.*/ boston|logan|fly|dl|746)"], "ground_fare"],	Ground Fare
[['(u_2 / much :advmod (u_1 / how) :amod-of (u_7 / cost))'], ["(.* / rent|get)","(.*/ dl|746)"], "ground_fare"],	Ground Fare
[['(u_1 / what :cop (u_2 / be) :nsubj (u_4 / cost))'], ["(.*/ flight|ticket|trip|fare)"], "ground_fare"],	Ground Fare
[['(u_1 / what :cop (u_2 / be) :nsubj (u_6 / rate))'], [], "ground_fare"],	Ground Fare
[['(u_2 / price :det (u_1 / what))'], [], "ground_fare"],	Ground Fare
[['(u_3 / list :aux (u_1 / can) :nsubj (u_2 / you) :obj (u_4 / cost))'], [], "ground_fare"],	Ground Fare
[['(u_2 / expensive :advmod (u_1 / how) :cop (u_3 / be))'], [], "ground_fare"],	Ground Fare
[['(u_2 / much :advmod (u_1 / how) :cop (u_3 / be))'], ["(.* / flight|ticket)"], "ground_fare"],	Ground Fare
[['(u_2 / list :discourse (u_1 / please) :obj (u_4 / price))'], [], "ground_fare"]	Ground Fare
[['(u_2 / number :compound (u_1 / flight) :nmod (u_4 / columbus|minneapolis :case (u_3 / from)))'], [], "flight_no"],	Flight Number
[['(u_149 / number :det (u_4 / the))'], ["(.*/ delta|worth|total|passenger|stop|aircraft|small)"], "flight_no"],	Flight Number
[['(u_1 / what :cop (u_2 / be) :nsubj (u_5 / number :det (u_3 / the) :compound (u_4 / flight)))'], [], "flight_no"],	Flight Number
[['(u_3 / have :aux (u_1 / may) :nsubj (u_2 / I) :obj (u_5 / listing :det (u_4 / a) :nmod (u_8 / number :case (u_6 / of) :compound (u_7 / flight))))'], [], "flight_no"],	Flight Number
[['(u_1 / list :obj (u_3 / number :det (u_2 / the) :nmod (u_5 / flight :case (u_4 / of) :acl (u_6 / arrive))))'], [], "flight_no"],	Flight Number
[['(u_5 / number :nsubj (u_1 / which) :cop (u_2 / be) :det (u_3 / the) :compound (u_4 / flight))'], [], "flight_no"]	Flight Number
[['(u_2 / many :advmod (u_1 / how) :amod-of (u_3 / seat|passenger|the|people))'], [], "capacity"],	Capacity
[['(u_1 / what :cop (u_2 / be) :nsubj (.* / capacity))'], [], "capacity"],	Capacity
[['(u_1 / list :obj (u_2 / number :nmod (u_4 / people :case (u_3 / of))))'], [], "capacity"]	Capacity

A.2 Examples

Table A.2: Examples on different Intent class.

Sentence[MS CNTK19]	RB	HYBRID	SVM	BERT	Intent
what is the arrival time in san francisco for the 755 am flight leaving washington	flight	flight	flight	flight_time	flight_time
how far is it from orlando airport to orlando	distance	distance	distance	distance	distance
what are the times that you have planes leaving from san francisco going to pittsburgh on july seventh	flight_time	flight_time	flight	flight_time	flight_time
how much does the limousine service cost within pittsburgh	ground_fare	ground_fare	ground_fare	airfare	ground_fare
what is the distance from los angeles international airport to los angeles	distance	distance	flight	distance	distance
what city is the airport mco in	city	city	ground_service	city	city
how much does it cost to rent a car in tacoma	ground_service	ground_service	ground_fare	airfare	ground_fare
on united airlines give me the flight times from boston to dallas	flight_time	flight_time	flight	flight_time	flight_time
what are the schedule of flights from boston to san francisco for august first	flight_time	flight_time	flight	flight_time	flight_time
flight numbers from columbus to minneapolis tomorrow	flight_no	flight_no	flight	flight_time	flight_no
where is mco	city	city	city	airline	city
what is the flight schedule of the f28 from pittsburgh to baltimore	flight_time	flight_time	flight	flight_time	flight_time
show me times for flights from san francisco to atlanta		flight	flight	flight_time	flight_time
i would like the time of all flights from san francisco to pittsburgh on sunday	flight	flight	flight	flight_time	flight_time
tell me distance from orlando airport to the city	distance	distance	flight	distance	distance
what are the costs of car rental in dallas	ground_service	ground_service	ground_fare	ground_service	ground_fare
could you give me the schedule of flights for american and delta to dfw on august fifteenth	flight_time	flight_time	flight	flight_time	flight_time
please list the flight times from pittsburgh to newark	flight_time	flight_time	flight	flight_time	flight_time
how far is downtown from the airport in dallas	distance	distance	distance	distance	distance
please list the flight times from boston to pittsburgh	flight_time	flight_time	flight	flight_time	flight_time
please list the flight schedule from baltimore to san francisco on friday nights	flight_time	flight_time	flight	flight_time	flight_time

how much does it cost to get downtown from the atlanta airport by limousine	ground_fare	ground_fare	airfare	airfare	ground_fare
i 'm trying to find the flight number from a flight from orlando to cleveland on us air and it arrives around 10 pm	flight	flight	flight	flight_time	flight_no
how long does it take to get from atlanta airport into the city of atlanta	distance	distance	flight	distance	distance
what times on wednesday could i take a plane from denver to oakland	flight_time	flight_time	flight	flight_time	flight_time
please tell me the times of the flights between boston and baltimore	flight	flight	flight	flight_time	flight_time
flight numbers from minneapolis to long beach on june twenty six	flight_no	flight_no	flight	flight_time	flight_no
show me times for coach flights between boston and baltimore on wednesday		flight	flight	flight_time	flight_time
show me the schedule for airlines leaving pittsburgh going to san francisco for next monday	flight_time	flight_time	flight	flight_time	flight_time
where is general mitchell international located	city	city	city	airline	city
what is the departure time of the latest flight of united airlines from denver to boston	flight	flight	flight	flight_time	flight_time
how long does it take to get from kansas city to st. paul	distance	distance	flight	quantity	distance
please show me the return flight number from toronto to st. petersburg	flight	flight	flight	flight_time	flight_no
what are the rental car rates in san francisco	ground_service	ground_service	ground_fare	ground_fare	ground_fare
what is the cost of the air taxi operation at philadelphia international airport	ground_service	ground_service	ground_service	ground_service	ground_fare
how much is a limousine between dallas fort worth international airport and dallas	ground_fare	ground_fare	airfare	airfare	ground_fare
where is general mitchell international located	city	city	city	airline	city
please list the flight times from pittsburgh to newark	flight_time	flight_time	flight	flight_time	flight_time
what is the distance between pittsburgh airport and downtown pittsburgh	distance	distance	ground_service	distance	distance
what times does continental depart from boston to san francisco	flight_time	flight_time	flight	flight_time	flight_time
what time does flight aa 459 depart	flight_time	flight_time	flight	flight_time	flight_time
what is the flight number for the continental flight which leaves denver at 1220 pm and goes to san francisco	flight	flight	flight	flight_time	flight_no

what is the number of first class flights on american airlines	flight_no	flight_no	flight	quantity	flight_no
please list the flight times from newark to boston	flight_time	flight_time	flight	flight_time	flight_time
what is the minimum connection time for houston intercontinental	flight_time	flight_time	flight	flight_time	flight_time
show me the cities served by nationair	city	city	city	city	city
how long does it take to fly from boston to atlanta	distance	distance	flight	quantity	distance
now i 'd like a schedule for the flights on tuesday morning from oakland no from dallas fort worth to atlanta	flight	flight	flight	flight_time	flight_time
what is the earliest departure time from boston to denver	flight_time	flight_time	flight	flight_time	flight_time
what price is a limousine service in boston	ground_fare	ground_fare	ground_service	airfare	ground_fare
where is lester pearson airport	city	city	city	airport	city
how much would car rental cost in atlanta	ground_service	ground_fare	ground_fare	airfare	ground_fare
please list the flight times for boston to pittsburgh	flight_time	flight_time	flight	flight_time	flight_time
show me the flight schedule from pittsburgh to san francisco	flight_time	flight_time	flight	flight_time	flight_time
what time does the tuesday morning 755 flight leaving washington arrive in san francisco	flight	flight	flight	flight_time	flight_time
how far is the airport from san francisco	distance	distance	distance	distance	distance
i would like a schedule of flights from san francisco to boston on wednesday	flight_time	flight_time	flight	flight_time	flight_time
what is the distance from la guardia to new york 's downtown	distance	distance	flight	distance	distance
can you list costs of denver rental cars	ground_service	ground_service	ground_fare	airfare	ground_fare
show me all the cities that midwest express serves	city	city	flight	city	city
may i have a listing of flight numbers from columbus ohio to minneapolis minnesota on monday	flight_no	flight_no	flight	flight_time	flight_no
how far from the airport in the dallas fort worth airport is dallas	distance	distance	distance	distance	distance
how much is the ground transportation between atlanta and downtown	airfare	airfare	ground_service	ground_service	ground_fare
how far is the airport from downtown pittsburgh	distance	distance	distance	distance	distance
what time are the flights leaving from denver to pittsburgh on july seventh	flight	flight	flight	flight_time	flight_time

i would like a schedule of flights from denver to san francisco on tuesday	flight_time	flight_time	flight	flight_time	flight_time
list the number of flights arriving in dallas fort worth from boston before noon	flight_no	flight_no	flight	quantity	flight_no
show me the cities served by canadian airlines international	city	city	city	city	city
what are the rental car rates in dallas	ground_service	ground_service	ground_fare	ground_fare	ground_fare
when does continental fly from philadelphia to denver on sundays		flight	flight	flight	flight_time
what time are the flights from baltimore to san francisco	flight	flight	flight	flight_time	flight_time
what price is a limousine service to new york 's la guardia	ground_fare	ground_fare	airfare	airfare	ground_fare
please give me the flight times the morning on united airlines for september twentieth from philadelphia to san francisco	flight_time	flight_time	flight	flight_time	flight_time
what 's the schedule of flights from atlanta to boston on august first	flight_time	flight_time	flight	flight_time	flight_time
what is the total schedule for delta 's flights to all airports	flight_time	flight_time	flight	flight_time	flight_time
how expensive is the san francisco limousine service	ground_fare	ground_fare	ground_fare	quantity	ground_fare
i would like the time your earliest flight from washington to philadelphia	flight_time	flight_time	flight	flight_time	flight_time
what is the cost of limousine service in philadelphia	ground_fare	ground_fare	ground_fare	ground_service	ground_fare
how far is it from salt lake city airport to salt lake city	distance	distance	distance	distance	distance
how long does it take to get from denver to oakland	distance	distance	flight	quantity	distance
what is the distance from san francisco international airport to san francisco	distance	distance	flight	distance	distance
what is the schedule of flights from boston to denver next monday	flight_time	flight_time	flight	flight_time	flight_time
is bwi washington	city	city	city	airline	city
what time does the flight leave denver going to san francisco on continental airlines	flight	flight	flight	flight_time	flight_time
what is the schedule for flights between pittsburgh and boston on the evening of july ninth	flight_time	flight_time	flight	flight_time	flight_time
what is delta 's schedule of morning flights to atlanta		flight	flight	flight_time	flight_time
what time zone is denver in	city	city	ground_service	flight_time	city

what is american 's schedule of morning flights to atlanta		flight	flight	flight_time	flight_time
what is the schedule of ground transportation from washington airport into downtown	ground_service	ground_service	ground_service	ground_service	flight_time
i would like the flight number and the time for the cheapest fare that is the least expensive first class fare from san francisco to pittsburgh leaving after 8 pm monday night	flight_time	flight_time	airfare	flight_time	flight_no
show me city served both by nationair and canadian airlines international	city	city	city	city	city
what cities are served by canadian airlines international	city	city	city	city	city
which is the flight number for the us air flight from philadelphia to boston is it 279 or is it 137338	flight_no	flight_no	flight	flight_time	flight_no
what time does twa depart from boston to go to san francisco	flight_time	flight_time	flight	flight_time	flight_time
what time does the earliest flight which goes from atlanta to denver leave	flight	flight	flight	flight_time	flight_time
i would like the evening schedule of flights from san francisco to washington	flight_time	flight_time	flight	flight_time	flight_time
please give me the flight times i would like to fly from boston to baltimore in the morning before 8		flight	flight	flight_time	flight_time
which cities does united airlines service	city	city	flight	city	city
what cities does continental service	city	city	flight	city	city
what are the cities that american airlines serves	city	city	flight	city	city
what is the flight number of the earliest flight between boston and washington dc	flight	flight	flight	flight_time	flight_no
what are the cities served by delta airlines	city	city	city	city	city
what times does the late afternoon flight leave from washington for denver	flight	flight	flight	flight_time	flight_time
what time are flights from denver to san francisco on continental airlines	flight	flight	flight	flight_time	flight_time
how long is a trip from philadelphia airport to downtown philadelphia	distance	distance	flight	quantity	distance

are there any other cities that i can fly from boston to dallas through that i can get a flight earlier than 1017 in the morning	flight	flight	flight	airline	city
list departure times from denver to philadelphia which are later than 10 o'clock and earlier than 2 pm		flight	flight	flight_time	flight_time
i would like the time of your earliest flight in the morning from philadelphia to washington on american airlines	flight_time	flight_time	flight	flight_time	flight_time
what is the cost of limousine service at logan airport	ground_service	ground_service	ground_service	ground_service	ground_fare
please list the flight times from newark to boston	flight_time	flight_time	flight	flight_time	flight_time
i want to know the time of the latest flight i can take from washington to san francisco where i can get a dinner meal	flight	flight	flight	flight_time	flight_time
what are the flight numbers of the flights which go from san francisco to washington via indianapolis	flight	flight	flight	flight_time	flight_no
what time is the last flight from washington to san francisco	flight	flight	flight	flight_time	flight_time
what is the distance from toronto international airport to toronto	distance	distance	flight	distance	distance
what are the rental car rates in dallas	ground_service	ground_service	ground_fare	ground_fare	ground_fare
which cities are serviced by both american and delta airlines	city	city	flight	airline	city
all right give me the flight times in the morning on september twentieth from pittsburgh to san francisco	flight_time	flight_time	flight	flight_time	flight_time
can you tell me the time a flight would leave from atlanta to boston in the afternoon	flight	flight	flight	flight_time	flight_time
what city is mco	city	city	city	city	city
what is the distance from boston airport to boston	distance	distance	flight	distance	distance
how far is oakland airport from downtown	distance	distance	distance	distance	distance
how long is the flight from atlanta to san francisco at noon on november seventh	flight	flight	flight	quantity	distance
please list the prices for a rental car in pittsburgh	ground_service	ground_service	ground_service	airfare	ground_fare

List of Figures

List of Tables

List of Algorithms

Acronyms

ATIS Airline Travel Information Systems. 1–3, 8, 12, 17, 23

BERT Bidirectional Encoder Representations from Transformers. xiii, 1–4, 6, 23, 28, 29, 33, 36–38, 40–43, 45–47, 49, 51–53, 56, 59, 61, 62

DL Deep Learning. 1, 35

FN False Negative. 20, 38

FP False Positive. 38

GPU Graphical Processing Unit. 2

HYBRID SVM-RBS Hybrid SVM-Rule Based System. 36–38, 41–44, 46, 47, 49, 51, 53, 56, 59, 61, 62

ML Machine Learning. 1, 5, 19, 24, 35

NLP Natural Language Processing. 5, 6, 9, 11, 24, 26, 29

NLU Natural Language Understanding. 6

RBF Radial Basis Function. 26

RBS Rule Based System. 20, 36–38, 40–45, 50, 51, 53, 61, 62

SVM Support Vector Machines. xiii, 1–4, 6, 23–27, 30, 31, 33, 35–38, 41–44, 46, 47, 49, 52, 53, 61, 62

TP True Positive. 20, 38

UD Universal Dependencies. 3, 8–10, 12, 14, 15, 75, 77

UI User Interface. 17, 19, 75

XAI Explainable Artificial Intelligence. 3, 16, 20

Bibliography

[Bha05] Sanjiv Bhatia. Regular expressions. *Computer Apex*, 01 2005.

[Bra07] Max Bramer. *Principles of Data Mining*. 01 2007.

[CLZ16] Juan Cabral, Nadia Luczywo, and José Zanazzi. Scikit-criteria: Colección de métodos de análisis multi-criterio integrado al stack científico de python. 09 2016.

[CZW19] Qian Chen, Zhu Zhuo, and Wen Wang. Bert for joint intent classification and slot filling. *arXiv preprint arXiv:1902.10909*, 2019.

[DCLT18] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. 2018.

[DVK17] Finale Doshi-Velez and Been Kim. Towards a rigorous science of interpretable machine learning, 2017.

[EG12] Martin Erwig and Rahul Gopinath. Explanations for Regular Expressions. In David Hutchison, Takeo Kanade, Josef Kittler, Jon M. Kleinberg, Friedemann Mattern, John C. Mitchell, Moni Naor, Oscar Nierstrasz, C. Pandu Rangan, Bernhard Steffen, Madhu Sudan, Demetri Terzopoulos, Doug Tygar, Moshe Y. Vardi, Gerhard Weikum, Juan De Lara, and Andrea Zisman, editors, *Fundamental Approaches to Software Engineering*, volume 7212, pages 394–408. Springer Berlin Heidelberg, Berlin, Heidelberg, 2012.

[ENCS19] Haihong E, Peiqing Niu, Zhongfu Chen, and Meina Song. A Novel Bi-directional Interrelated Model for Joint Intent Detection and Slot Filling. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, Florence, Italy, 2019. Association for Computational Linguistics.

[FSV01] Pasquale Foggia, Carlo Sansone, and Mario Vento. An improved algorithm for matching large graphs. 01 2001.

[GGH+18] Chih-Wen Goo, Guang Gao, Yun-Kai Hsu, Chih-Li Huo, Tsung-Chieh Chen, Keng-Wei Hsu, and Yun-Nung Chen. Slot-Gated Modeling for Joint Slot Filling and Intent Prediction. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, New Orleans, Louisiana, June 2018. Association for Computational Linguistics.

[gita] GitHub - adaamko/POTATO: XAI based human-in-the-loop framework for automatic rule-learning. — github.com. `https://github.com/adaamko/POTATO`. [Accessed 22-Feb-2023].

[gitb] GitHub - recski/tuw-nlp: NLP @ TU Wien — github.com. `https://github.com/recski/tuw-nlp`. [Accessed 22-Feb-2023].

[GMMRMSUL] Ismael Garrido-Muñoz, Arturo Montejo-Ráez, Fernando Martínez-Santiago, and L. Alfonso Ureña-López. A survey on bias in deep NLP. 11(7):3184.

[GN00] Emden R. Gansner and Stephen C. North. An open graph visualization system and its applications to software engineering. *Software Practice & Experience*, 30(11):1203–1233, September 2000.

[Goo20] Michael Wayne Goodman. Penman: An open-source library and tool for AMR graphs. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 312–319, Online, July 2020. Association for Computational Linguistics.

[HSS08] Aric A. Hagberg, Daniel A. Schult, and Pieter J. Swart. Exploring network structure, dynamics, and function using networkx. In Gaël Varoquaux, Travis Vaught, and Jarrod Millman, editors, *Proceedings of the 7th Python in Science Conference*, pages 11 – 15, Pasadena, CA USA, 2008.

[HTTC+16] Dilek Hakkani-Tür, Gokhan Tur, Asli Celikyilmaz, Yun-Nung Chen, Jianfeng Gao, Li Deng, and Ye-Yi Wang. Multi-Domain Joint Semantic Frame Parsing Using Bi-Directional RNN-LSTM. In *Interspeech 2016*. ISCA, September 2016.

[IKT05] Emmanouil Ikonomakis, Sotiris Kotsiantis, and V. Tampakas. Text classification using machine learning techniques. *WSEAS transactions on computers*, 4:966–974, 08 2005.

[Joa99] Thorsten Joachims. Text categorization with support vector machines. Technical report, Universität Dortmund, October 1999.

[Kec05] V. Kecman. Support Vector Machines – An Introduction. In Janusz Kacprzyk and Lipo Wang, editors, *Support Vector Machines: Theory and Applications*, volume 177, pages 1–47. Springer Berlin Heidelberg, Berlin, Heidelberg, April 2005.

[KGIR22] Ádám Kovács, Kinga Gémes, Eszter Iklódi, and Gábor Recski. Potato: Explainable information extraction framework. In *Proceedings of the 31st ACM International Conference on Information Knowledge Management*, CIKM '22, page 4897–4901, New York, NY, USA, 2022. Association for Computing Machinery.

[lab] sklearn.preprocessing.LabelEncoder — scikit-learn.org. `https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.LabelEncoder.html#sklearn.preprocessing.LabelEncoder`. [Accessed 23-Feb-2023].

[LG] Han Liu and Alexander Gegov. Rule based systems and networks: Deterministic and fuzzy approaches. In *2016 IEEE 8th International Conference on Intelligent Systems (IS)*, pages 316–321. IEEE.

[LL16] Bing Liu and Ian Lane. Attention-Based Recurrent Neural Network Models for Joint Intent Detection and Slot Filling, September 2016. arXiv:1609.01454 [cs].

[LLQ18] Changliang Li, Liang Li, and Ji Qi. A Self-Attentive Model with Gate Mechanism for Spoken Language Understanding. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, Brussels, Belgium, 2018. Association for Computational Linguistics.

[MMS$^+$] Ninareh Mehrabi, Fred Morstatter, Nripsuta Saxena, Kristina Lerman, and Aram Galstyan. A survey on bias and fairness in machine learning.

[MS] Christopher D. Manning and Hinrich Schütze. Cambridge, Massachusetts.

[MS CNTK19] MS CNTK. Atis (airline travel information systems), 2019. data retrieved from Kaggle, `https://www.kaggle.com/code/siddhadev/atis-dataset-from-ms-cntk/data`.

[MYJ18] Ramesh Mande, Kalyan Chakravarti Yelavarti, and G JayaLakshmi. Regular Expression Rule-Based Algorithm for Multiple Documents Key Information Extraction. In *2018 International Conference on Smart Systems and Inventive Technology (ICSSIT)*, pages 262–265, Tirunelveli, India, December 2018. IEEE.

[MZ09] Marcelo Mendoza and Juan Zamora. Identifying the intent of a user query using support vector machines. pages 131–142, 08 2009.

[niv] Universal dependencies 2.3.

[OB] Ahmed Hamza Osman and Omar Mohammed Barukub. Graph-based text representation and matching: A review of the state of the art and future challenges. 8:87562–87583.

[PGM+19] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019.

[Pra12] Ashis Pradhan. Support vector machine-a survey. *IJETAE*, 2, 09 2012.

[PVG+12] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Andreas Müller, Joel Nothman, Gilles Louppe, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Édouard Duchesnay. Scikit-learn: Machine Learning in Python. 2012.

[pypa] pip — pypi.org. `https://pypi.org/project/pip/`. [Accessed 22-Feb-2023].

[pypb] xpotato — pypi.org. `https://pypi.org/project/xpotato/`. [Accessed 22-Feb-2023].

[QZZ+20] Peng Qi, Yuhao Zhang, Yuhui Zhang, Jason Bolton, and Christopher D. Manning. Stanza: A Python natural language processing toolkit for many human languages. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, 2020.

[RLKH] Gábor Recski, Björn Lellmann, Adam Kovacs, and A. Hanbury. Explainable rule extraction via semantic graphs.

[Sas07] Yutaka Sasaki. The truth of the f-measure. *Teach Tutor Mater*, 01 2007.

[Sch] Hugo Schnack. Bias, noise, and interpretability in machine learning. In *Machine Learning*, pages 307–328. Elsevier.

[sci] sklearn.svm.SVC — scikit-learn.org. `https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html`. [Accessed 23-Feb-2023].

[SHRJ21] Yilin Shen, Yen-Chang Hsu, Avik Ray, and Hongxia Jin. Enhancing the generalization for intent classification and out-of-domain detection in SLU. *CoRR*, abs/2106.14464, 2021.

[spa] spaCy · Industrial-strength Natural Language Processing in Python — spacy.io. `https://spacy.io/`. [Accessed 23-Feb-2023].

[SS89] Jungyun Seo and Robert F. Simmons. Syntactic graphs: A representation for the union of all ambiguous parse trees. *Comput. Linguist.*, 15(1):19–32, mar 1989.

[str] Streamlit • The fastest way to build and share data apps — streamlit.io. `https://streamlit.io/`. [Accessed 22-Feb-2023].

[SYG19] Prashanth Gurunath Shivakumar, Mu Yang, and Panayiotis Georgiou. Spoken Language Intent Detection using Confusion2Vec. 2019.

[WDS+19] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. HuggingFace's Transformers: State-of-the-art Natural Language Processing. 2019.

[WM10] Wes McKinney. Data Structures for Statistical Computing in Python. In Stéfan van der Walt and Jarrod Millman, editors, *Proceedings of the 9th Python in Science Conference*, pages 56 – 61, 2010.

[WSJ18] Yu Wang, Yilin Shen, and Hongxia Jin. A Bi-Model Based RNN Semantic Frame Parsing Model for Intent Detection and Slot Filling. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, New Orleans, Louisiana, 2018. Association for Computational Linguistics.

[YA20] Reda Yacouby and Dustin Axman. Probabilistic extension of precision, recall, and f1 score for more thorough evaluation of classification models. In *Proceedings of the First Workshop on Evaluation and Comparison of NLP Systems*, pages 79–91, Online, November 2020. Association for Computational Linguistics.

[ZBY07] Nivio Ziviani and Ricardo Baeza-Yates. *String Processing and Information Retrieval, 14th International Symposium, SPIRE 2007, Santiago, Chile, October 29-31, 2007, Proceedings*, volume 4726. 01 2007.

[ZLD+19] Chenwei Zhang, Yaliang Li, Nan Du, Wei Fan, and Philip Yu. Joint Slot Filling and Intent Detection via Capsule Neural Networks. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, Florence, Italy, 2019. Association for Computational Linguistics.