RAMAIAH Institute of Technology

Department of Computer Science & Engineering

Program: Bachelor of Engineering in Computer Science & Engineering

A Project Report on

Out Domain Utterance detection

Submitted in partial fulfillment of the requirements for the course

By

Ananya Muralidhar	1MS18CS018
Angel Paul	1MS18CS019
Jeevan Kumar	1MS19ET042
Sreyas Acharya	1MS19CS116
Harsh Dutta Tewari	1MS20CS050

Mr. Sathwick Mahadeva
Mr. Kavit Gangar
Staff Engineer and Senior Software Engineer

Dr.S.Rajarajeswari
Associate Professor

M S RAMAIAH INSTITUTE OF TECHNOLOGY
(Autonomous Institute, Affiliated to VTU)
BANGALORE-560054
www.msrit.edu
2021

M. S. RAMAIAH INSTITUTE OF TECHNOLOGY, BANGALORE – 560 054
(Autonomous Institute, Affiliated to VTU)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

RAMAIAH
Institute of Technology

Department of Computer Science & Engineering

CERTIFICATE

Certified that the project work entitled "Out Domain Utterance Detection" carried out by Jeevan Kumar (1MS19ET042) and Shreyas Acharya(1MS19CS116) and Angel Paul (1MS18CS019), Ananya Muralidhar(1MS18CS018) and Harsh Dutta Tewari (1MS20CS050) are bonafide students of M.S.Ramaiah Institute of Technology Bengaluru in partial fulfillment for the award of Bachelor of Engineering in Computer Science and Engineering of the Visvesvaraya Technological University, Belgavi during the year 2018-19. It is certified that all corrections/suggestions indicated for Internal Assessment have been incorporated in the report deposited in the department library.

The project report has been approved as it satisfies the academic requirements in respect of Project work prescribed for the said Degree.

Project Guide
DR.S.RAJARAJESWARI

Head of the Department
DR.ANNAPURNA P. PATIL

External Examiners

Name of the Examiners: **Signature with Date**

1. Sathwick Mahadeva

2. Kavit Gangar

The certificate of Participation from Samsung should be attached here.

DECLARATION

We, hereby, declare that the entire work embodied in this project report has been carried out by us at M.S.Ramaiah Institute of Technology, Bengaluru, in collaboration with Samsung R&D Institute India-Bangalore (SRI-B) , under the supervision of Mr. Sathwick Mahadeva (Staff Engineer), Mr. Kavit Gangar(Senior Software Engineer) and **Dr.S.Rajarajeswari, Associate Professor,** Dept of CSE. This report has not been submitted in part or full for the award of any diploma or degree of this or to any other university.

Signature
Angel Paul
1MS18CS019

Signature
Ananya Muralidhar
1MS18CS018

Signature
Jeevan Kumar
1MS19ET042

Signature
Shreyas Acharya
1MS19CS116

Signature
Harsh Dutta Tewari
1MS20CS050

ACKNOWLEDGEMENT

We take this opportunity to express our gratitude to the people who have been instrumental in the successful completion of this project. We would like to express our profound gratitude to the Management and **Dr. N.V.R Naidu** Principal, M.S.R.I.T, Bengaluru for providing us with the opportunity to explore our potential.

We extend our heartfelt gratitude to our beloved **Dr.Annapurna Patil**, HOD, Computer Science and Engineering, for constant support and guidance.

We wholeheartedly thank our Industry project guide **Mr. Sathwick Mahadeva (Staff Engineer), Mr. Kavit Gangar(Senior Software Engineer) and Dr.S.Rajarajeswari, Associate Professor** for providing us with the confidence and strength to overcome every obstacle at each step of the project and inspiring us to the best of our potential. We also thank her for her constant guidance, direction and insight during the project.

This work would not have been possible without the guidance and help of several individuals who in one way or another contributed their valuable assistance in preparation and completion of this study.

Finally, we would like to express sincere gratitude to all the teaching and non-teaching faculty of CSE Department, our beloved parents, seniors and my dear friends for their constant support during the course of work.

ANGEL PAUL

ANANYA MURALIDHAR

JEEVAN KUMAR

SHREYAS ACHARYA

HARSH DUTTA TEWARI

Abstract

Dialog systems must be able to discern whether an input sentence is in-domain (ID) or out-of-domain (OD) to provide an acceptable user experience (OOD). We assume that only ID sentences are available as training data since gathering enough OOD sentences in an unbiased manner is a time-consuming and tedious task. We initially devised a few ways to avoid out-of-domain datasets and solely utilize in-domain datasets for training. Using a multi-Class model was one of the solutions. In-domain data were used to categorize each speech into its specific class in the multi-class model. Any utterances that could not be classified were designated as out-domain utterances. After comprehensive testing of the multi-class models, a number of barriers were discovered, especially as the number of classes rose. Additionally, issues were caused by the first dataset due to the presence of the same utterances in both in-domain and out-domain datasets. As a result, a Binary Classification model was considered. Both in-domain and out-domain data were employed in the binary classification model at first, later switching to using just in-domain data for training and out-domain data for testing. A new dataset was selected resulting in a higher accuracy with the binary model as the new dataset was more extensive, clean, and consistent without any redundant utterances. This work introduces a unique approach that encodes phrases in a low-dimensional continuous vector space while emphasizing characteristics distinguishing ID instances from OOD situations. We examined our technique by empirically comparing it to state-of-the-art methods; The LSTM-Autoencoder model was the best binary classification method as it obtained the highest accuracy in all tests.

.

TABLE OF CONTENTS

1. Introduction

1.1 General Introduction

Most dialog systems except for general-purpose dictation systems, function across specific domains which the users aren't often aware of. Domain of the utterance by the user is a field the utterance belongs to. The user is expected to give out utterances of domains involved in the service during conversation with a dialogue system. The system responds with utterance not comprehensive when the user tells an utterance that doesn't belong to any of the service domains of the system. These kinds of utterances are referred to as out-of-domain utterances. In more formal terms, in-domain (ID) utterances are those that belong to one of the service domains and accordingly the service is provided, and out-of-domain (OOD) are those that don't belong to any of the service domains. If an utterance belongs to any service domain, it will still be an OOD if the requested function is not delivered by the system. For example, in a service domain 'tv channels' with one function to 'play abc channel', then the question 'what program is currently playing' will not be recognized by the system. Such OOD utterances should be predicted and detected by the spoken language systems.

It is critical to recognise OOD utterances in order to improve the usability of the system, it will allow users to decide whether to retry the current job after confirming that its in-domain, or to discontinue as the utterance would be OOD. For example, if the system wasn't able to process an in-domain utterance and then recognizes it when the user rephrases the utterance, the same can't be the case when an out-of-domain utterance is encountered. The system will not be able to handle the request regardless of it being rephrased. It's considerably more difficult to detect out-of-domain utterances for virtual assistant systems than it is to design chatbots for a specific domain. Unlike domain-specific chatbots, which may rely only on gathering out-of-domain data iteratively and improving overall performance, virtual assistants are often unable to use the customized OOD datasets. This can be due to the fact that these assistants may originate from various domains and have varying distributions. Customized intents classification models would not be able to make use of large numbers of OOD samples, particularly if compute resources are constrained. As a result, text from the out-of-domain utterance pool must be

down-sampled. Moreover, because out-of-domain utterances from production environments are unlikely to be detected by models during development and training, classifiers may struggle to distinguish out-of-domain utterances from in-domain utterances, and results may differ considerably in each round of testing. To capture out-of-domain utterances, systems must be able to predict as well as detect them. To predict out-of-domain utterances, the language model must have some coverage margin, like statistical language models instead of grammar-based models, and a methodology is required to detect out-of-domain utterances.

In this paper, we propose the usage of deep learning classification models to classify all the utterances only using in-domain datasets into either in-domain or out-domain utterances. The data to these models are converted to N dimensional vectors using models like OneHot embedding, Glove, BERT and Word2Vec. We compare performances of different multi-class classification models like LSTM and CNN and binary classification models like LSTM-Autoencoder, Bidirectional LSTM, One class SVM, GAN and others.

1.2 Problem Statement

One significant problem with the classifier is identification of out-of-domain utterances - utterances which doesn't belong to any of the supported capsules. Creating a model capable of identifying a user's utterance as out-of-domain so that Bixby can take appropriate action for it. To train out of domain utterances we need a rich/big data set of out of domain utterances which is difficult to get.

1.3 Objective of the Project

- Understanding the implementation of existing solutions to identify their drawbacks
- Propose a method to solve the problem of identifying out of domain utterances which will solve the limitations of the existing solutions.
- Compare the performance of the proposed AI/ML model with existing solutions
- Evaluate if the proposed model can be integrated with bixby.

1.4 Project Deliverables:

- Data set preparation/collection suitable for the task.
- Developing ML/DL model for accurately identifying out of domain utterances using in domain utterances to train the model.

1.5 Current Scope

The current systems adopt dataset interpolation, and thus uses the existing dataset for domain detection for study of OOD detection. It treats each service domain as OOD. The performance is measured using EER (equal error rate value). Construction of a large dataset for the dialogue system is required. It extracts the lexical, syntactic and semantic features to train a binary SVM classifier using a large number of random web-search queries and VPA utterances from multiple domains.

1.6 Future Scope

In future work, we also intend to investigate approaches to improve discrimination of OOD utterances and erroneously recognized in-domain utterances.

Identifying out domain utterances from the speaker can reduce the number of times the speaker rephrases the query to VPA. The current systems can be improved by identifying ways to distinguish between OOD utterances and incorrectly identified in-domain utterances. Since it is more difficult to produce longer documents for OOD detection, the concept in classification issues for longer text can be investigated. The possibility of giving additional elements of the task-oriented conversation system, such the dialogue state monitoring or dialogue management modules, the capacity to recognise OOD inputs merits further study.

2.PROJECT ORGANIZATION

2.1 Software Process Models -Agile Model

Agile is best suited to manage systems that involve variability. It also supports quick modifications in the project's scope and directions based on the changes required. This approach helps to deliver quick and more effective results and provides long-term project maintenance. The Agile model promotes flexibility and provides rapid improvement of projects in a consistent manner. The agile model promotes collaborative working. It notifies ways to improve the collaborations, test them and measure its success keeping in mind the primary focus. It is best suited when the project is broken into smaller pieces, which are then prioritized by the team in terms of importance.

2.2 Roles and responsibilities

Name	Role and responsibilities
Ananya Muralidhar, Jeevan Kumar, Shreyas Acharya, Angel Paul	Work on various Binary and multi class ML models to detect out-domain utterance.
Harsh Dutta Tewari, Ananya Muralidhar, Jeevan Kumar	Presentations, technology, documentation, paper.

3. LITERATURE SURVEY

3.1 Introduction

The current systems adopt dataset interpolation, and thus uses the existing dataset for domain detection for study of OOD detection. It treats each service domain as OOD. The performance is measured using EER (equal error rate value). Construction of a large dataset for the dialogue system is required. It extracts the lexical, syntactic and semantic features to train a binary SVM classifier using a large number of random web-search queries and VPA utterances from multiple domains.

3.2 Related works

Year	**Research Paper**	**Methodology**	**Drawbacks/ Future scope**
2017	Neural sentence embedding using only in-domain sentences for out-of-domain sentence detection in dialog systems.	Used unlabelled text to pre-train word representations followed by domain category analysis to train neural sentence embedding. These are used to train the autoencoder for OOD detection.	Accuracy of the autoencoder + DC-LSTM two channels can be improved.
2006	Out-of-Domain Utterance Detection Using Classification Confidences of Multiple Topics	Uses Topic Classification to classify each topic with a confidence score and performs in-domain verification based on the confidence scores to detect ODD.	To implement the proposed framework, an adequate set of pre-defined topic classes are required.
2014	Detecting Out-Of-Domain Utterances Addressed to a Virtual Personal Assistant	The task is to build a classifier to detect orphan utterances using large amounts of utterances used to build domain specific models and random keyword queries hitting web search.	Detecting uncovered utterances addressed to a VPA is surprisingly a hard task. This task is more related to

			addressee detection or dialog act tagging than domain detection task.
2019	Survey on Out-Of-Domain Detection for Dialog Systems	The deleted interpolation is adopted, making the existing datasets for domain detection available to the study of OOD detection. It treats each service domain as OOD. The performance is measured using the equal error rate(EER) value.	The datasets have a common limitation that they are not for development of the dialog system. It is necessary to construct and share a large dataset for the dialog system.
1998	Confidence Scoring For Speech Understanding Systems	Uses an automatic labeling algorithm based on a semantic frame comparison between recognized and transcribed orthographies. Then exploring the recognition-based features along with semantic, linguistic, and application-specific features for utterance rejection. Discriminant analysis is used in an iterative process to select the best set of classification features for the utterance rejection sub-system.	The analysis of the user's behavior to rejected utterances suggests that more informative feedback is needed in order to prevent error spirals. Therefore the intend is to add word level confidence measures to detect early problems with certain content words.

REFERENCES

[1] Ryu, S., Kim, S., Choi, J., Yu, H., & Lee, G. G. (2017). Neural sentence embedding using only in-domain sentences for out-of-domain sentence detection in dialog systems. *Pattern Recognition Letters*, *88*, 26-32.

[2] Lane, I., Kawahara, T., Matsui, T., & Nakamura, S. (2006). Out-of-domain utterance detection using classification confidences of multiple topics. *IEEE Transactions on Audio, Speech, and Language Processing*, *15*(1), 150-161.

[3] Tür, G., Deoras, A., & Hakkani-Tür, D. (2014, September). Detecting out-of-domain utterances addressed to a virtual personal assistant. In *Interspeech* (pp. 283-287). Pao, C., Schmid, P., & Glass, J. R. (1998, December).

[4] Jeong, Y. S., & Kim, Y. M. (2019). Survey on Out-Of-Domain Detection for Dialog Systems. *Journal of Convergence for Information Technology*, *9*(9), 1-12.

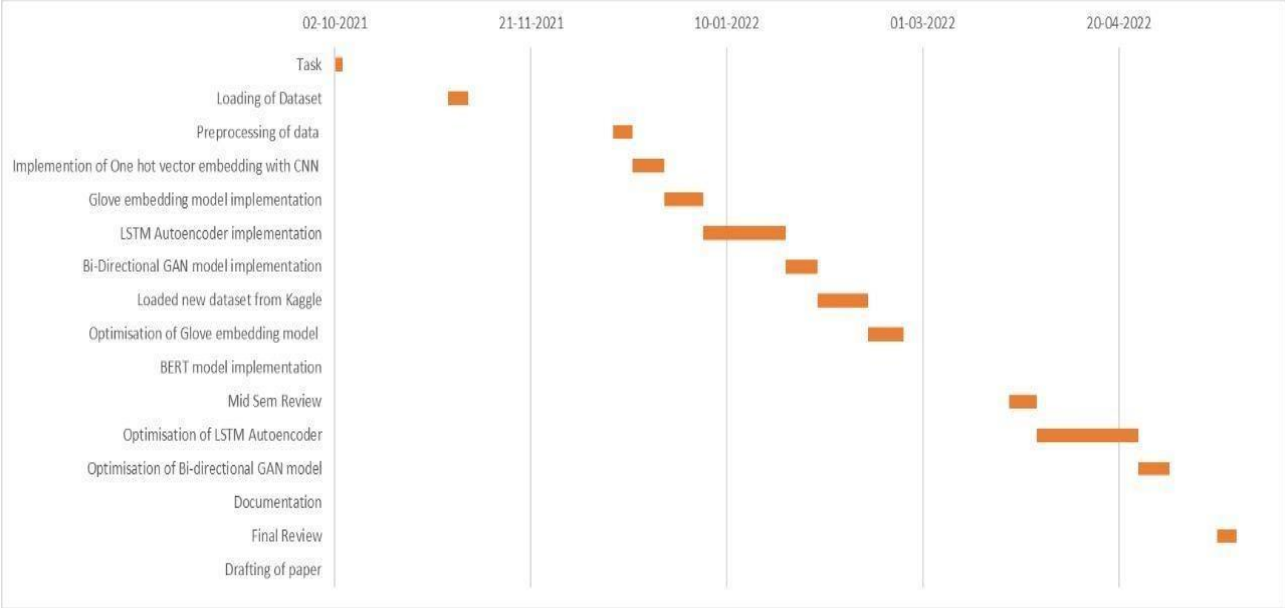
[5] Confidence scoring for speech understanding systems. In *ICSLP* (pp. 815-818).

3.3 Conclusion

With several aliases, such as anomaly detection, one-class classification, open-set recognition, or novelty detection, the topic of OOD detection has been studied in a variety of situations. Conventional techniques have produced significant outcomes in low-dimensional areas, and some of these techniques have also been used with NLU systems. For OOD identification, some modern neural networks just need in domain data. The majority of these techniques use the threshold-based methodology, and other ways of computing the detection scores have been developed. Modeling the probability density, calculating reconstruction losses, using classifier ensembles, using Bayesian models, relying on distances to nearest neighbors, or even explicitly learning a detection score are examples of popular techniques. However, the majority of these approaches lack the computing power to fully benefit from unlabeled data to enhance OOD detection performance, whether it be during training or inference.

4.PROJECT MANAGEMENT PLAN

4.1 Schedule of the Project (Represent it using Gantt Chart)



4.2 Risks identified with Fog Computing

Users of spoken dialogue systems (SDS) expect high-quality interactions across a wide range of subject matter. However, implementing SDS capable of responding to every conceivable user utterance in an informative way is a challenge. Multi-domain SDS must necessarily identify and deal with out-of-domain (OOD) utterances to generate appropriate responses. Users do not always know in advance what domains the SDS can handle which might lead to misclassification. Due to the complexity of the model, there is a high requirement for computational power; hence, it may be difficult for it to run on devices efficiently, potentially causing performance issues.

5. SOFTWARE REQUIREMENT SPECIFICATIONS

5.1 Product Overview

Deep learning classification models are employed to categorize all of the utterances into either in-domain or out-domain utterances using just in-domain datasets. These models use techniques like One Hot vector embedding, Glove, BERT, and Word2Vec to transform the data into N-dimensional vectors. The performances of various binary classification models such as LSTM-Autoencoder, Bidirectional LSTM, One-class SVM, GAN, and others as well as multi-class classification models such as LSTM and CNN were examined.

5.2 External Interface Requirements

5.2.1 User interface.

Since our model is based on a back-end Integration, it does not have direct interaction with the user. In this implementation, users interact with only one application at a time in a manner determined by their user roles. When necessary, the application alerts other applications of the important user interaction details. Application-to-application interactions can be based on further sequential notifications.

5.2.2 Hardware Interface

The computational power of a high-performance Graphics Processing Unit (Tensor flow's GPU interface) was used to train the model utilizing CUDA Technology.

5.2.3 Software Interface

An array of built-in Python Libraries such as (Keras, Scikit Learn, pandas, NumPy, matplotlib etc) were used.

5.3 Functional Requirements

FR1: Processing power: Due to the complexity of the model, there is a high requirement for the computational power; The computational power of a high-performance Graphics Processing Unit is required to reduce the performance issues.

FR2: Classification: Categorizing all of the utterances into either in-domain or out-domain utterances using just in-domain datasets.

FR3: Performance: refers to the software's ability to meet time requirements.

FR4: Interoperability: refers to the degree to which two or more systems can exchange meaningful information through interfaces in a particular context. The medical area is characterized by having several environments with different systems, where information is generated in different formats. In certain situations, it is important that information created in one application can be used by others which are only able to manipulate a different format.

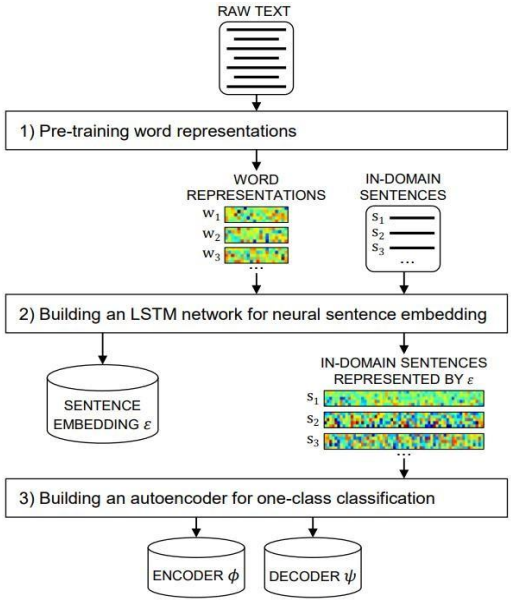
6.DESIGN

6.1 Introduction

The input data that is given to the model can be classified into 2 parts namely in-domain and out of domain data. In-domain (ID) utterances are those that belong to one of the service domains and accordingly the service is provided, and out-of-domain (OOD) are those that don't belong to any of the service domains. If an utterance belongs to any service domain, it will still be an OOD if the requested function is not delivered by the system. For example, in a service domain ‘SetAlarm’ with one function to ‘Set alarm for a PM’, then the question ‘What is the time now ’will not be recognized by the system. Such OOD utterances should be predicted and detected by the spoken language systems for better serviceability and good user experience as in case of OOD input, user can be prompted to retry or can be given a generic answer of a web search result rather than a totally out of context info.

In this paper, we propose the usage of deep learning classification models to classify all the utterances only using in-domain datasets into either in-domain or out-domain utterances. The data to these models are converted to N dimensional vectors using models like OneHot embedding, Glove, BERT and Word2Vec. We compare performances of different multi-class classification models like LSTM and CNN and binary classification models like LSTM-Autoencoder, Bidirectional LSTM, One class SVM, GAN and others.

6.2 Architecture Design

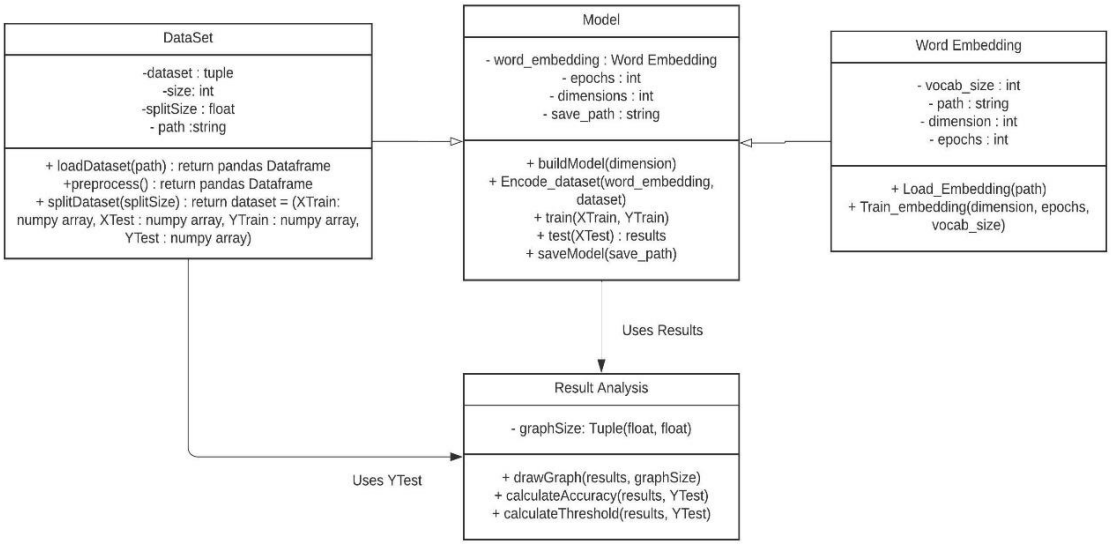


There are two approaches for this problem, a multi-class classification model and a binary classification model. In a multiclass classification model, we assume all the capabilities of a system to be a class. For example, “Set an alarm at 6:00 am” would be part of the ‘alarm’ class. “Call John mobile” would be part of the ‘phone’ class. Here ‘alarm’ and ‘phone’ would be the capabilities of a system and hence both these utterances would be classified as in-domain utterances. Each in-domain utterance would be assigned to a class, and the multi-class model would classify each utterance it receives to a certain class. When an out-domain utterance is passed to the same model, it would fail to classify it to any existing class, hence we would identify it as an out-domain utterance. We use confidence score or probabilities to determine if the utterance is not classifiable into any class. If N number of utterances are passed into the model, it will return N number probabilities, i.e., the probability of that particular utterance belonging to that particular class. If none of the N probabilities is greater that 0.70, then we say that the utterance does not belong to any class and therefore is an outdomain utterance. In a binary classification model, we assume all the in-domain utterances to be of a single in-domain class and all out-domain utterances to be of a single out-domain class.

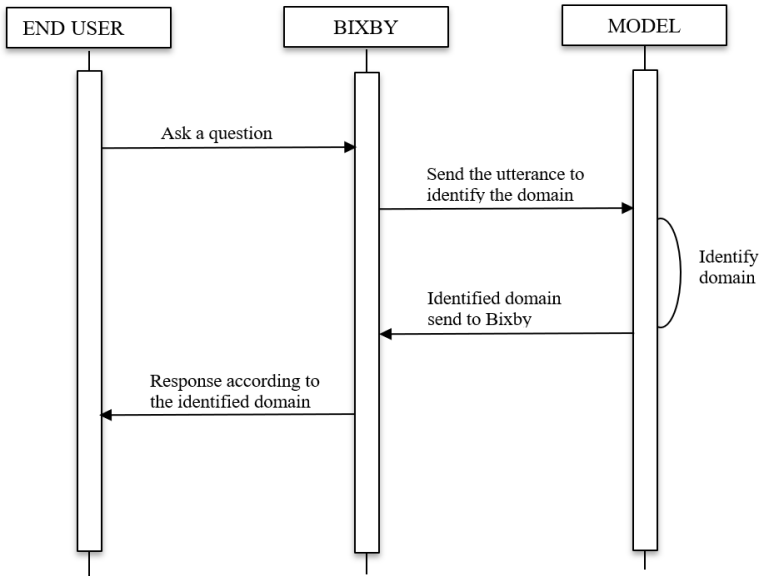
6.3 Graphical User Interface

The model is a backend integration model and thus, has no direct interaction with the user. In this implementation, users interact with only one application at a time according to their user roles. The other applications are notified of the significant aspects of the user interaction as necessary by this application. The interaction between the applications can be based on the further sequence of notifications.

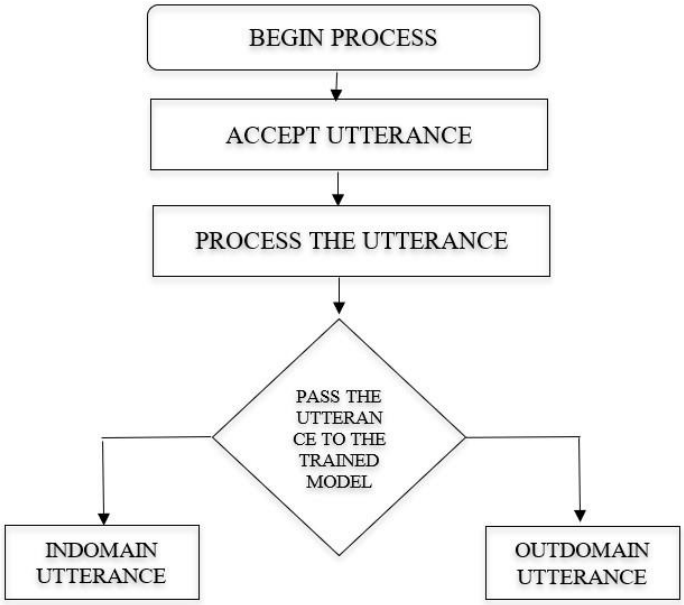
6.4 Class Diagram



6.5 Sequence Diagram



6.6 Data flow diagram



6.7 Conclusion

The OOD detection and proper classification of ID and OOD is an essential topic that is being researched upon. There are Conventional Techniques that can produce significant outcomes for low-dimensional areas and can be integrated with NLU. But on the other hand, we also have modern methods which can perform the OOD Identification efficiently and need the ID data only. Modelling the probability density, calculating reconstruction losses, using classifier ensembles, using Bayesian models, relying on distances to nearest neighbours, threshold-based methodologies are some of the ways these new methods rely upon. However, the majority of these approaches lack the computing power to fully benefit from unlabelled data to enhance OOD detection performance, whether it be during training or inference.

7. IMPLEMENTATION

7.1 Tools Introduction

7.1.1 TensorFlow

Tensorflow is a free and open-source software library for machine learning and artificial intelligence. Although it can be applied to many different tasks, deep neural network training and inference are given special attention. The Google Brain team created it for use within Google. With regard to this project, we constructed the neural networks using a few Tensorflow modules, including Dense, Dropout, Concatenate, Input, and a few activation functions.

7.1.2 Gensim

A modern statistical machine learning technique called Gensim is used for unsupervised topic modeling, document indexing, retrieval by similarity, and other NLP functionalities. The word embedding models for this project were created using the Word2Vec gensim module.

7.1.3 Scikit-learn

A free machine learning library for the Python programming language is called Scikit-learn. Support-vector machines, random forests, gradient boosting, k-means, and DBSCAN are just a few of the classification, regression, and clustering algorithms it includes. We tested the performance of a few built-in anomaly detection algorithms, including OneClassSVM and IsolationForest, for this project.

7.1.4 Keras

A Python interface for artificial neural networks is provided by the open-source software library known as Keras. The TensorFlow library interface is provided by Keras. Several built-in Keras modules were used to preprocess the dataset for this project.

7. 1. 5 Numpy

Numpy is open-source software for processing arrays. It offers a multidimensional array object with high performance as well as tools for

interacting with these arrays. It is the cornerstone Python package for scientific computing.

7.1.6 Matplotlib

For the Python programming language and its NumPy numerical mathematics extension, Matplotlib is a plotting library. For embedding plots into programs using all-purpose GUI toolkits like Tkinter, wxPython, Qt, or GTK, it offers an object-oriented API. The project's entire graph visualization was carried out with the aid of matplotlib modules like plot, hist, etc.

7.1.7 Pandas

Pandas is an open-source library designed primarily for working quickly and logically with relational or labeled data. It offers a range of data structures and operations for working with time series and numerical data.

7.2 Technology Introduction

7.2.1 Machine learning

An area of artificial intelligence (AI) called machine learning (ML) enables computers to "self-learn" from training data and get better over time without having to be explicitly programmed. Detecting patterns in data and learning from them allows machine learning algorithms to develop their own predictions.

In conventional programming, an engineer for computers creates a set of instructions that tell a machine how to change input data into the desired output. The majority of instructions follow an IF-THEN structure: when particular criteria are met, the program performs a particular action. In contrast, machine learning is a process that is automated and gives computers the ability to solve issues with little to no human involvement and make decisions based on prior experiences.

Types of Machine learning :

1. Supervised Learning techniques make predictions based on labeled training data. Input and the desired output are both included in each training sample. This

sample data is examined by a supervised learning algorithm, which then draws a conclusion.

2. Unsupervised learning techniques are given input data, but since the desired results are unknown, they must draw conclusions based on the available data. Clustering is one of the most popular unsupervised learning techniques.
3. Semi-Supervised Learning: A small number of the dataset for semi-supervised learning are labeled, while the majority are unlabeled. To make predictions about the unlabeled data, the model uses labeled data.
4. Reinforcement Learning models determine the best course of action to take in a particular circumstance. The machine picks the actions that result in the best solution or the highest reward after learning from its own mistakes.
5. Deep learning models can be fully supervised, partially supervised, unsupervised, or even a mix of all three. Neurons, which simulate how the human brain functions, are the foundation of deep learning. They are made up of many layers of connected neurons, which enables multiple systems to operate at once.

7.2.2 Natural Language Processing

Building computational algorithms to automatically analyze and represent human language is known as natural language processing (NLP). Numerous applications, including Google's robust search engine and, more recently, Amazon's voice assistant named Alexa, are made possible by NLP-based systems. NLP is helpful for teaching machines how to carry out difficult natural language-related tasks, like dialogue generation and machine translation.

Embeddings in Word The so-called distributional hypothesis states that words with similar contexts have similar meanings. This is the foundation for distributional vectors, also known as word embeddings. Word embeddings are pre-trained using a shallow neural network on a task where the goal is to predict a word based on its context. RNNs are specialized neural-based methods for processing sequential data that are efficient. The results of previous computations are used to conditionally apply computation to each instance of an input sequence by an RNN. A fixed-size vector of tokens that are sequentially (or one at a time) fed to a recurrent unit serves as a typical representation of these sequences.

The ability of an RNN to remember the outcomes of earlier computations and use that knowledge in the current computation is its key strength. RNN models can thus be used to represent context dependencies in inputs of any length in order to properly compose the input. RNNs have been used to research a variety of NLP tasks, including language modeling, image captioning, and machine translation, among others.

7.3 Overall view of the project in terms of implementation

In out-domain utterance detection, the primary task is to detect an utterance that is not within the capabilities of a system. Hence, it is efficient to see it immediately and prompt the user/client. Deep Learning is used to detect such an utterance, but the primary problem with this method is the lack of a consistent out-domain dataset. A deep learning classification model has to be created which classifies all utterances into an in-domain or an out-domain utterance, and this model has to be trained only using an in-domain dataset.

Approach :

There are two approaches to this problem, a multi-class classification model and a binary classification model.

In a multiclass classification model, we assume all the capabilities of a system to be a class. For example, “Set an alarm at 6:00 am” would be part of the ‘alarm’ class. “Call John mobile” would be part of the ‘phone’ class. Here ‘alarm’ and ‘phone’ would be the capabilities of a system and hence both these utterances would be classified as in-domain utterances. Each in-domain utterance would be assigned to a class, and the multi-class model would classify each utterance it receives to a certain class. When an out-domain utterance is passed to the same model, it would fail to classify it to any existing class, hence we would identify it as an out-domain utterance. We use confidence score or probabilities to determine if the utterance is not classifiable in any class. If N number of utterances are passed into the model, it will return N number probabilities, i.e., the probability of that particular utterance belonging to that particular class. If none of the N

probabilities is greater than 0.70, then we say that the utterance does not belong to any class and therefore is an out-domain utterance.

In a binary classification model, we assume all the in-domain utterances to be of a single in-domain class and all out-domain utterances to be of a single out-domain class. The model will classify the input utterance into either of the classes. However, training the model with only an in-domain dataset returned a sub-par accuracy (<40%), hence an LSTM-Autoencoder model is used to overcome this.

While the multiclass model performs well when the number of classes is below 30-40, the accuracy declines as we use it for 100+ classes. Due to this Binary classification is preferred, and since only an in-domain dataset has to be used we use an LSTM Autoencoder m

Preprocessing Data

The utterances used in this model are in the form of text. All the text is converted to N-dimensional vectors before passing it to train the model. To do this, various models like One Hot, Glove, BERT, and Word2Vec were used. As Word2Vec returned the highest accuracy, it was the optimal choice for this problem.

The vocabulary was obtained using only in-domain data. Each word was assigned a vector of 200 dimensions. The vectors are allotted in a way such that 2 words that are similar are closer to each other in the vector space. For example, ‘King’ is closer to ‘man’ than to ‘airplane’. After the vectors are assigned to every word in the utterance, it is pre-padded with ‘0’s to ensure all the utterances are of the same length

7.4 Explanation of Algorithm and Implementation of Modules

LSTM-Autoencoder :

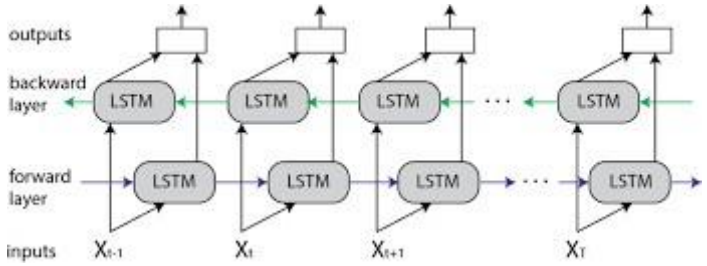
The used model resembles a binary classifier, however, uses only in domain utterances during the training phase. It contains two neural networks, the Bidirectional LSTM and the Autoencoder. The bidirectional LSTM plays the role of a neural sentence embedding

network that is to represent the input utterances as an n-dimensional vector space. And the autoencoder classifies the utterances as either in a domain or out domain data.

Before training the neural sentence embedding model the utterances are represented as a continuous vector space using various embedding techniques such as word2vec and glove.

Word embedding is a feature extraction technique that encodes the meaning of the word in a fixed dimensional vector space such that the words that are closer in the vector space are expected to be similar in meaning. Multiple dimensions of both the word embedding models were trained and tested and the optimal dimension which resulted in a consistently high accuracy was 200.

The Bidirectional LSTM Model is a recurrent neural network. A Recurrent Neural Network is a class of artificial neural networks where the output from the previous set is fed as input to the current step. It is this mechanism that gives an RNN its “memory” factor. Bi-LSTM (Bidirectional long short-term memory) is a model that consists of two LSTMs, one taking information from the forward direction, and the other in a backward direction which increases the amount of information available to the network.

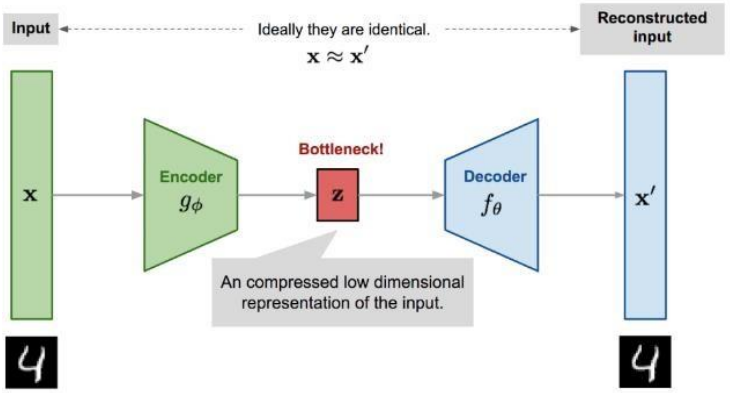


The model is trained with one assumption that all the data are labeled with specific classes despite the fact that they all belong to in domain class. For example: “book ticket” may belong to flight class and “order food” may belong to restaurant class, and these collectively belong to the in-domain class. This is done to preserve the domain-specific information. During the training phase, the model might encounter some rarely occurring words which hinder the fine-tuning of the model. A solution to this problem is to use two channels: a static and a non-static channel. The non-static channel is fine-tuned whereas

the static channel is not. To prevent overfitting, dropout layers are used. The dropout layer randomly sets input units to 0 with a rated frequency at each step during training time. The values in the last hidden layer, that is the concatenation of static and non-static layers, is used to represent the utterance.

Autoencoder is an unsupervised learning technique whose aim is to learn lower-dimensional representations for higher-dimensional data. The architecture consists of three parts:

1. Encoder: Compresses the input data into a lower-dimensional space.
2. Bottleneck: Part that contains the information regarding the compressed knowledge representation. It is the most important part of the network.
3. Decoder: Decompresses the data and reconstructs the data back to its original dimensional space.



The outputs from the Bidirectional LSTM are used to train the autoencoder. The encoder function, denoted by ϕ, maps the original data X, to a latent space F, which is present at the bottleneck. The decoder function, denoted by ψ, maps the latent space F at the bottleneck to the output. Finally, the reconstruction error is calculated, and if the error is less than the threshold the data is classified as domain data. else as out domain data.

BiGAN :

A Generative adversarial network (GAN) is a class of unsupervised techniques that involves learning the regularities or patterns in input data such that it can be later on used

to generate new examples that plausibly could have been drawn from the original dataset.

A GAN has two parts: a generative network and a discriminative network. A Generative network maps a fixed length random vector to a vector space of interest. The discriminator is a simple classifier that tries to distinguish between the real data and the data created by the generator. Both networks compete against each other in the training phase. The Discriminator tries to minimize its loss and the generator tries to maximize the discriminator's loss. The training process can be mathematically described by the formula below:

$$\min_G \max_D V(D,G)$$

$$V(D,G) = \mathbb{E}_{x \sim p_{data}(x)}[\log D(x)] + \mathbb{E}_{z \sim p_z(z)}[\log(1 - D(G(z))]$$

Training a GAN has the following two parts:

1. The discriminator is trained while the generator is idle. In this phase, the network is only forward propagated. The discriminator is trained on real data and the fake data to see if it can correctly classify them.
2. The Generator is trained while the Discriminator is idle. The results from the trained Discriminator are used to train the Generator.

For Anomaly Detection, a variant of GAN is used known as the BiGAN. A BiGAN includes an Encoder network which enables the model to map the real space to the latent space. The Encoder is structured as the inverse of the generator.

A Bi-LSTM along with word-embedding techniques like word2ved, the glove is used to pre-process the data. The training process is similar to that of regular GAN. But unlike regular GAN where the discriminator considers only the inputs, the Discriminator in a BiGAN also considers the latent representation. Finally, the Encoder and the Generator are used to find the Reconstruction error, based on which the data is classified as in domain or Out domain.

8. TESTING

8.1 Introduction

Testing is the process of evaluating a system and its components with the intent to find whether it satisfies the specified requirements or not. In simple words, testing is executing a system in order to identify any gaps, errors, or missing requirements contrary to the actual requirements.

Since machine learning is about learning the behavior of the data, testing involves validating the consistency of the model's logic and desired behavior.

8.2 Test cases

Test Utterance	Description	Expected Result
Setting an alarm at a specific time.	The user desires to set an alarm for a particular time, for example seven in the morning. In this situation, the input utterance might be something like "Set an alarm for 7 am." or "Set an alarm for seven in the morning".	Setting an alarm is regarded as a function that a personal assistant is capable of performing, thus the model categorizes it as an in-domain utterance and instructs a series of commands to be executed.
Dialing a number.	The user wants to make a call to someone. In this situation, saying "Call XYX" or "Call 080-123-456" would be appropriate.	One of a personal assistant's functions is making calls. As a result, the model categorizes it as an in-domain utterance and directs the system to look through its contacts or dial a phone number.
Ordering food.	The user wants to place an online meal order from an area restaurant. The appropriate phrase in this scenario may be "Order one pizza	One of the functions of a personal assistant is ordering food, thus the model classifies this as an in-domain utterance and tells the system to do

	from XYZ Restaurant."	so.
Obtaining facts about a well-known individual.	The user may be interested in details about a famous person, such as their occupation, birth date, and accomplishments. The appropriate question might be, "What NGO is XYZ a part of?" or "When was XYZ born."	Such information must be searched for online, but searching online is not a feature of a personal assistant. As a result, the model categorizes this as an out-of-domain utterance and either shows a message advising the user of this or refers them to a search engine like Google.

9 RESULTS & PERFORMANCE ANALYSIS

9.1 Result Snapshots

Although the binary models returned very high accuracies, they required both in-domain and out-domain data to train them. In the event where only in-domain data was used, the accuracy was drastically lower and cannot be used. Hence, the Bidirectional LSTM Autoencoder was the model that showed the most promise, but the BiGAN model also returned a good accuracy.

The Autoencoder with Bidirectional LSTM (two channels) was the most accurate in out-domain sentence detection using only in-domain data. The model had an accuracy of 79.15 % with a threshold of 1.81 as shown in Figure 1. The BiGAN gave an accuracy of 76.47% with a threshold of .1195 as shown in Figure 2.

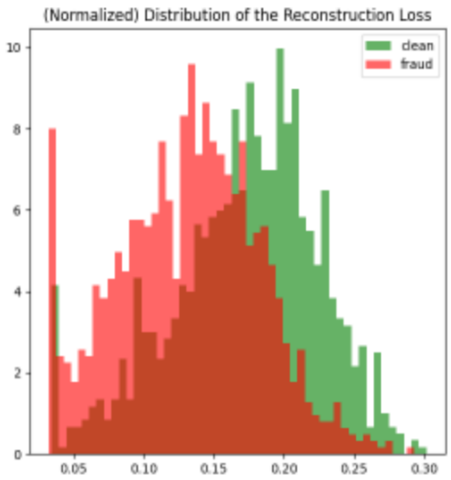


Figure 1:Reconstruction error using the Autoencoder

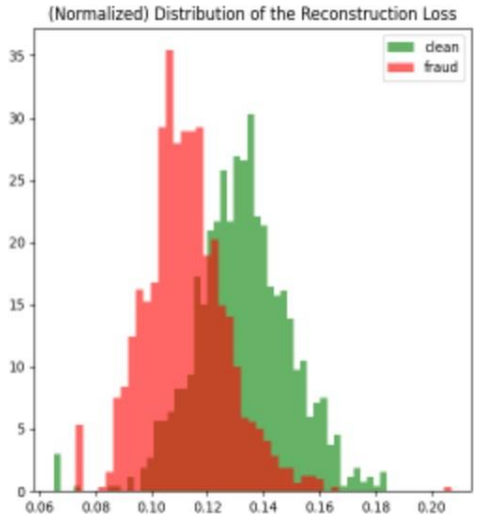


Figure 2: Reconstruction error using the GAN

10. CONCLUSION & SCOPE FOR FUTURE WORK

Although the multi-class model returns a high accuracy on average, the same drastically decreases when the number of classes/labels inside the in-domain data increases. A confidence score of 75-80% was used to decide if the utterance is in-domain or out-domain. Due to the variation inaccuracies, it was not an ideal choice for a complex voice assistant with a broad spectrum of features. Binary classifiers returned the highest accuracy with certain models. However, this was achievable only while passing both in-domain and out-domain data. The models which were trained solely using in-domain data returned low accuracy ranging from 40-60% and hence would not be an optimal solution either. The Autoencoder model is one we conclude to be optimal at this stage returning an accuracy of 79.15%. Here, the in-domain data and out-domain data were distinguished using the reconstruction error method. The reconstruction errors in the autoencoder were low for ID sentences but high for OOD sentences on average. This model uses only in-domain data to train the model, ideal considering the lack of proper out-domain datasets.

11. References

[1] Khan, S., & Madden, M. (2014). One-class classification: Taxonomy of study and review of techniques. *The Knowledge Engineering Review, 29*(3), 345-374. doi:10.1017/S026988891300043X

[2] A. B. Nassif, M. A. Talib, Q. Nasir and F. M. Dakalbab, "Machine Learning for Anomaly Detection: A Systematic Review," in IEEE Access, vol. 9, pp. 78658-78700, 2021, doi: 10.1109/ACCESS.2021.3083060.

[3] Taeshik Shon, Jongsub Moon, A hybrid machine learning approach to network anomaly detection, Information Sciences, Volume 177, Issue 18, 2007, Pages 3799-3821, ISSN 0020-0255,
https://doi.org/10.1016/j.ins.2007.03.025.

[4] Li, D., Chen, D., Jin, B., Shi, L., Goh, J., Ng, SK. (2019). MAD-GAN: Multivariate Anomaly Detection for Time Series Data with Generative Adversarial Networks. In: Tetko, I., Kůrková, V., Karpov, P., Theis, F. (eds) Artificial Neural Networks and Machine Learning – ICANN 2019: Text and Time Series. ICANN 2019. Lecture Notes in Computer Science(), vol 11730. Springer, Cham. https://doi.org/10.1007/978-3-030-30490-4_56

[5] Seonghan Ryu, Seokhwan Kim, Junhwi Choi, Hwanjo Yu, Gary Geunbae Lee, Neural sentence embedding using only in-domain sentences for out-of-domain sentence detection in dialog systems, Pattern Recognition Letters, Volume 88, 2017, Pages 26-32, ISSN 0167-8655,
https://doi.org/10.1016/j.patrec.2017.01.008.

[6] Houssam Zenati, Chuan Sheng Foo, Bruno Lecouat, Gaurav Manek, Vijay Ramaseshan Chandrasekhar, Efficient GAN-Based Anomaly Detection, arXiv:1802.06222

[7] Sarah M. Erfani, Sutharshan Rajasegarar, Shanika Karunasekera, Christopher Leckie, High-dimensional and large-scale anomaly detection using a linear one-class SVM with deep learning,

Pattern Recognition, Volume 58, 2016, Pages 121-134, ISSN 0031-3203, https://doi.org/10.1016/j.patcog.2016.03.028.

[8] Dang, W., Zhou, B., Wei, L., Zhang, W., Yang, Z., Hu, S. (2021). TS-Bert: Time Series Anomaly Detection via Pre-training Model Bert. In: Paszynski, M., Kranzlmüller, D., Krzhizhanovskaya, V.V., Dongarra, J.J., Sloot, P.M.A. (eds) Computational Science – ICCS 2021. ICCS 2021. Lecture Notes in Computer Science(), vol 12743. Springer, Cham. https://doi.org/10.1007/978-3-030-77964-1_17

[9] Z. Chen, C. K. Yeo, B. S. Lee and C. T. Lau, "Autoencoder-based network anomaly detection," 2018 Wireless Telecommunications Symposium (WTS), 2018, pp. 1-5, doi:10.1109/WTS.2018.8363930.

[10] Meira, J.; Carneiro, J.; Bolón-Canedo, V.; Alonso-Betanzos, A.; Novais, P.; Marreiros, G. Anomaly Detection on Natural Language Processing to Improve Predictions on Tourist Preferences. Electronics 2022, 11, 779. https://doi.org/10.3390/electronics11050779

[11] Guansong Pang, Chunhua Shen, Longbing Cao, and Anton Van Den Hengel. 2021. Deep Learning for Anomaly Detection: A Review. ACM Comput. Surv. 54, 2, Article 38 (March 2022), 38 pages. https://doi.org/10.1145/3439950

[12] Kwon, D., Kim, H., Kim, J. et al. A survey of deep learning-based network anomaly detection. Cluster Comput 22, 949–961 (2019). https://doi.org/10.1007/s10586-017-1117-8

Out domain utterance detection

Jeevan Kumar, Shreyas Acharya, Angel Paul, Ananya Muralidhar, Harsh Dutta Tewari
Ramaiah Institute of Technology
Department of Computer Science and Engineering,
MSRIT Post, Bangalore 560054

Abstract: Dialog systems must be able to discern whether an input sentence is in-domain (ID) or out-of-domain (OD) to provide an acceptable user experience (OOD). We assume that only ID sentences are available as training data since gathering enough OOD sentences in an unbiased manner is a time-consuming and tedious task. We initially devised a few ways to avoid out-of-domain datasets and solely utilize in-domain datasets for training. Using a multi-Class model was one of the solutions. In-domain data were used to categorize each speech into its specific class in the multi-class model. Any utterances that could not be classified were designated as out-domain utterances. After comprehensive testing of the multi-class models, a number of barriers were discovered, especially as the number of classes rose. Additionally, issues were caused by the first dataset due to the presence of the same utterances in both in-domain and out-domain datasets. As a result, a Binary Classification model was considered. Both in-domain and out-domain data were employed in the binary classification model at first, later switching to using just in-domain data for training and out-domain data for testing. A new dataset was selected resulting in a higher accuracy with the binary model as the new dataset was more extensive, clean, and consistent without any redundant utterances. This work introduces a unique approach that encodes phrases in a low-dimensional continuous vector space while emphasizing characteristics distinguishing ID instances from OOD situations. We examined our technique by empirically comparing it to state-of-the-art methods; The LSTM-Autoencoder model was the best binary classification method as it obtained the highest accuracy in all tests.

Keywords: Out-domain utterance, In-domain utterance LSTM-Autoencoder, BERT, Glove, Word2Vec, Glove, GAN, Bidirectional LSTM.

I. INTRODUCTION

Most dialog systems except for general-purpose dictation systems, function across specific domains which the users aren't often aware of. Domain of the utterance by the user is a field the utterance belongs to. The user is expected to give out utterances of domains involved in the service during conversation with a dialogue system. The system responds with utterance not comprehensive when the user tells an utterance that doesn't belong to any of the service domains of the system. These kinds of utterances are referred to as out-of-domain utterances. In more formal terms, in-domain (ID) utterances are those that belong to one of the service domains and accordingly the service is provide, and out-of-domain (OOD) are those that don't' belong to any of the service domains. If an utterance belongs to any service domain, it will still be an OOD if the requested function is not delivered by the system. For example, in a service domain 'tv channels' with one function to 'play abc channel', then the question 'what program is currently playing' will not be recognized by the system. Such OOD utterances should be predicted and detected by the spoken language systems.

It is critical to recognise OOD utterances in order to improve the usability of the system, it will allow users to decide whether to retry the current job after confirming that its in-domain, or to discontinue as the utterance would be OOD. For example, if the system wasn't able to process an in-domain utterance and the recognizes it when the user rephrases the utterance, but same can't be the case when an out-of-domain utterance is encountered. The system will not be able to handle the request regardless of it being rephrased. It's considerably more difficult to detect out-of-domain utterances for virtual assistant systems than it is to design chatbots for a specific domain. Unlike domain-specific chatbots, which may rely only on gathering out-of-domain data iteratively and improving overall performance, virtual assistants are often unable to use the customized OOD datasets. This can be due to the fact that these assistants may originate from various domains and have varying distributions. Customized intents classification models would not be able to make use of large number of OOD samples, particularly if compute resources are constrained. As a result, text from the out-of-domain utterance pool must be down-sampled. Moreover, because out-of-domain utterances from production environments are unlikely to be detected by models during development and training, classifiers may struggle to distinguish out-of-domain utterances from in-domain utterances, and results may differ considerably in each round of testing. To capture out-of-domain utterances, systems must be able to predict as well as detect them. To predict out-of-domain utterances, the language model must have some coverage margin, like statistical language models instead of grammar-based models, and a methodology is required to detect out-of-domain utterances.

In this paper, we propose the usage of deep learning classification models to classify all the utterances only using in-domain datasets into either in-domain or out-domain utterances. The data to these models are converted to N dimensional vectors using models like OneHot embedding, Glove, BERT and Word2Vec. We compare performances of different multi-class classification models like LSTM and CNN and binary classification models like LSTM-Autoencoder, Bidirectional LSTM, One class SVM, GAN and others.

II. Related works

[1] This paper, gives an overview of the previous studies of OOD detection in terms of three point of view: dataset, feature, and method. It adopts dataset interpolation, and thus uses the existing dataset for domain detection for study of OOD detection. It treats each service domain as OOD. The performance is measured using EER (equal error rate value). Construction of a large dataset for the dialogue system is required. [2] It is important to identify this rejected OOD utterances so that the VPA can work as it is intended. To tackle the problem, the paper extracts the lexical, syntactic and semantic features to train a binary SVM classifier using a large number of random web-search queries and VPA utterances from multiple domains. It leaves one domain out and checks the model's accuracy when dataset used is having some unseen queries. Results suggest that the use of such structured features provides quite high accuracy especially when test domain has a little resemblance with the existing domain. [3] This paper proposes an OOD detection framework which applies a linear discriminant model to perform in-domain verification by using Classification Confidence Scores of various topics. The verification model is high portable and can be trained by using a combination of deleted interpolation of in-domain data and minimum-classification-error training. The proposed approach achieves an absolute reduction in OOD detection errors its performance is equivalent to a model trained by both ID and ODD data. This framework can also be applied to the "machine-aided-dialogue" corpus to achieve a furthermore reduction in EER. [4] This research investigates the use of utterance-level features for confidence scoring. It demonstrates a novel automatic labelling algorithm based on a semantic frame comparison between recognized and transcribed orthographies. Experiments show that the proposed methodology can correctly reject over 60% of incorrectly understood utterances while accepting 98% of all correctly understood utterances. [5] This paper proposes a new neural sentence embedding method that represents sentences in a low-dimensional continuous vector space that focuses on aspects that distinguish ID cases from OOD. It proposes a methodology in which a large unlabelled text is used to pre-train word representations and then, the domain-category analysis is used to train the Neural Sentence Embedding. The sentence representations that were learned are used to train an autoencoder aimed at OOD sentence detection. It is proven that this method is quite efficient and competes with the state-of-the-art methods in its accuracy.

III. Theory / Calculation / Methodology

In out-domain utterance detection, the primary task is to detect an utterance that is not within the capabilities of a system. Hence, it is efficient to detect it immediately and prompt the user/client. Deep Learning is used to detect such an utterance, but the primary problem with this method is the lack of a consistent out-domain dataset. A deep learning classification model has to be created which classifies all utterances into an in-domain or an out-domain utterance, and this model has to be trained only using an in-domain dataset.

Approach

There are two approaches for this problem, a multi-class classification model and a binary classification model.

In a multiclass classification model, we assume all the capabilities of a system to be a class. For example, "Set an alarm at 6:00 am" would be part of the 'alarm' class. "Call John mobile" would be part of the 'phone' class. Here 'alarm' and 'phone' would be the capabilities of a system and hence both these utterances would be classified as in-domain utterances. Each in-domain utterance would be assigned to a class, and the multi-class model would classify each utterance it receives to a certain class. When an out-domain utterance is passed to the same model, it would fail to classify it to any existing class, hence we would identify it as an out-domain utterance. We use confidence score or probabilities to determine if the utterance is not classifiable into any class. If N number of utterances are passed into the model, it will return N number probabilities, i.e., the probability of that particular utterance belonging to that particular class. If none of the N probabilities is greater that 0.70, then we say that the utterance does not belong to any class and therefore is an out-domain utterance.

In a binary classification model, we assume all the in-domain utterances to be of a single in-domain class and all out-domain utterances to be of a single out-domain class. The model will classify the input utterance into either of the classes. However, training the model with only in-domain dataset returned a sub-par accuracy (<40%), hence an LSTM-Autoencoder model is used to overcome this.

While the multiclass model performs well when the number of classes are below 30-40, the accuracy declines as we use it for 100+ classes. Due to this Binary classification is preferred, and since only in-domain dataset has to be used we use a LSTM Autoencoder model.

Preprocessing Data

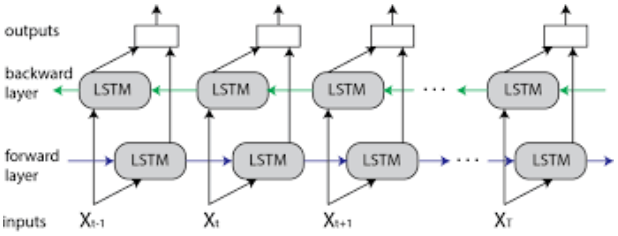
This model uses utterances that are textual in nature. Prior to being sent to train the model, all text is transformed into N-dimensional vectors. Several models, including One Hot, Glove, BERT, and Word2Vec, were used to achieve this. Word2Vec was the best option for this issue because it gave the best accuracy results.

Only data within the domain was used to obtain the vocabulary. A 200-dimensional vector was given to each word. Two words that are similar are placed closer to one another in the vector space by the way the vectors are assigned. For instance, "King" is more akin to "man" than "aeroplane." The length of each utterance is ensured by padding it with zeros.

LSTM-Autoencoder:

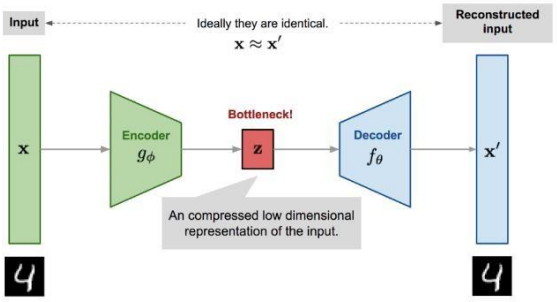
Although the used model only uses in-domain utterances during the training phase, it resembles a binary classifier. The Bidirectional LSTM and the Autoencoder are two of the neural networks that are used. A neural sentence embedding network uses the bidirectional LSTM to represent the input utterances as an n-dimensional vector

space. And the autoencoder divides the utterances into data that is in domain and data that is out of domain.
The utterances are represented as a continuous vector space prior to training the neural sentence embedding model using a variety of embedding techniques, including word2vec and glove.
Word embedding is a feature extraction method that encodes a word's meaning in a fixed-dimensional vector space, with the expectation that words that are close to one another in the vector space will have similar meanings. Multiple word embedding model dimensions were tested and trained, and the best dimension that consistently produced high accuracy was 200.
The Bidirectional LSTM Model is a recurrent neural network. A Recurrent Neural Network are a class of artificial neural network where the output from previous set is fed as input to the current step. It is this mechanism that gives a RNN it's "memory" factor. In order to increase the amount of information available to the network, the Bi-LSTM (Bidirectional long short-term memory) model consists of two LSTMs, one of which takes information from the forward direction and the other from the backward direction.



Despite the fact that all of the data belong to the same domain class, the model is trained under the assumption that each set of data is labelled with a specific class. For instance, while "order food" and "book ticket" may belong to the restaurant class and the flight class, respectively, these terms collectively fall under the in-domain class. To protect the domain-specific information, this is done. The model may run into some uncommon words during the training phase, which prevents it from being fine-tuned. Use of two channels—a static channel and a non-static channel—is the solution to this issue. In contrast to the static channel, the non-static channel has been fine-tuned. Dropout layers are used to avoid overfitting. During training, the dropout layer randomly sets the input units to 0 with a rate at each step. The utterance is represented by the values in the final hidden layer, which is created by concatenating the static and non-static layers.
Autoencoder is an unsupervised learning technique whose aim is to learn lower-dimensional representation for a higher-dimensional data. The architecture consists of three parts:

1. Encoder: Reduces the dimensions of the input data by compressing it.
2. Bottleneck: Area where data on the compressed knowledge representation is located. Of the network, it is the most crucial.
3. Decoder: Rebuilds the data into its original dimensional space after decompressing it.



The autoencoder is trained using the Bidirectional LSTM's outputs. The bottleneck's latent space F is mapped to the original data X by the encoder function, denoted by ϕ . The decoder function, symbolised by ψ, converts the latent space F at the bottleneck into output. After calculating the reconstruction error, the data is categorised as in-domain data if the error is below a threshold. else as our domain information.

BiGAN :
A class of unsupervised technique known as a generative adversarial network (GAN) involves learning the regularities or patterns in input data so that it can later be used to produce new examples that could have been reasonably derived from the original dataset.
A GAN has two parts: a generative network and a discriminative network.
A Generative network maps a fixed length random vector to a vector space of interest. The discriminator is a simple classifier that tries to distinguish between the real data and the data created by the generator. In the training phase, the two networks are in competition with one another. The generator tries to increase the discriminator's loss while the discriminator tries to minimise its loss. The following formula can be used to mathematically describe the training process:

$$\min_G \max_D V(D,G)$$

$$V(D,G) = \mathbb{E}_{x\sim p_{data}(x)}[\log D(x)] + \mathbb{E}_{z\sim p_z(z)}[\log(1 - D(G(z))]$$

Training a GAN has the following two parts:

a. The discriminator is trained while the generator is idle. In this phase, the network is only forward propagated. The discriminator is trained on real data and the fake data to see if it can correctly classify them.
b. The Generator is trained while the Discriminator is idle. The results from the trained Discriminator are used to train the Generator.

A GAN variant known as the BiGAN is used for anomaly detection. An encoder network, which is a component of a BiGAN, allows the model to map the real space to the latent space. The generator's opposite in structure, the encoder.

To pre-process the data, a Bi-LSTM and word-embedding methods like word2ved and glove are used. The training procedure is comparable to that of a standard GAN. But in a BiGAN, the discriminator also takes into account the latent representation, unlike in a regular GAN where the discriminator only takes into account the inputs. The data is classified as being in the domain or being outside the domain based on the Reconstruction error that is found using the Encoder and the Generator.

IV. RESULT AND DISCUSSIONS

Although the binary models returned very high accuracies, they required both in-domain and out-domain data to train them. In the event where only in-domain data was used, the accuracy was drastically lower and cannot be used. Hence, the Bidirectional LSTM Autoencoder was the model that showed the most promise, but the BiGAN model also returned a good accuracy.

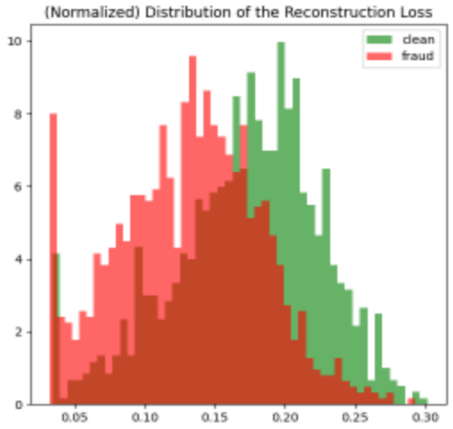


Figure 1: Reconstruction error using Autoencoder

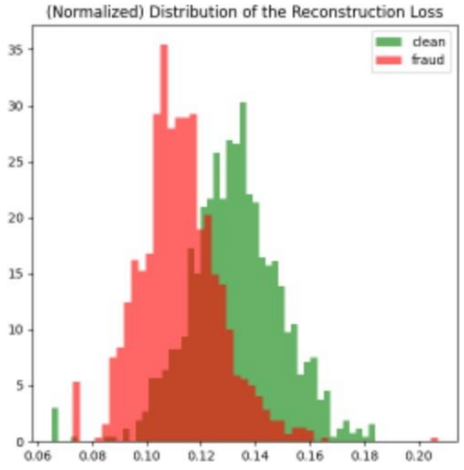


Figure 2: Reconstruction error using BiGAN

The Autoencoder with Bidirectional LSTM (two channels) was the most accurate in out- domain sentence detection using only in domain data. The model had an accuracy of 79.15 % with a threshold of 1.81 as shown in Figure 1. The BiGAN gave an accuracy of 76.47% with a threshold of .1195 as shown in Figure 2.

V. CONCLUSION

Although the multi-class model returns a high accuracy on average, the same drastically decreases when the number of classes/labels inside the in-domain data increases. A confidence score of 75-80% was used to decide if the utterance is in-domain or out-domain. Due to the variation inaccuracies, it was not an ideal choice for a complex voice assistant with a broad spectrum of features. Binary classifiers returned the highest accuracy with certain models. However, this was achievable only while passing both in-domain and out-domain data. The models which were trained solely using in-domain data returned low accuracy ranging from 40-60% and hence would not be an optimal solution either. The Autoencoder model is one we conclude to be optimal at this stage returning an accuracy of 79.15%. Here, the in-domain data and out-domain data were distinguished using the reconstruction error method. The reconstruction errors in the autoencoder were low for ID sentences but high for OOD sentences on average. This model uses only in-domain data to train the model, ideal considering the lack of proper out-domain datasets.

VI. REFERENCES

[1] Jeong, Y. S., & Kim, Y. M. (2019). Survey on Out-Of-Domain Detection for Dialog Systems. *Journal of Convergence for Information Technology*, *9*(9), 1-12.

[2] Tür, G., Deoras, A., & Hakkani-Tür, D. (2014, September). Detecting out-of-domain utterances addressed to a virtual personal assistant. In *Interspeech*

[3] Lane, I., Kawahara, T., Matsui, T., & Nakamura, S. (2006). Out-of-domain utterance detection using classification confidences of multiple topics. *IEEE Transactions on Audio, Speech, and Language Processing*, *15*(1), 150-161.

[4] Pao, C., Schmid, P., & Glass, J. R. (1998, December). Confidence scoring for speech understanding systems. In *ICSLP* (pp. 815-818).

[5] Ryu, S., Kim, S., Choi, J., Yu, H., & Lee, G. G. (2017). Neural sentence embedding using only in-domain sentences for out-of-domain sentence detection in dialog systems. *Pattern Recognition Letters*, *88*, 26-32.

[Samsung PRISM] End Review Report

Bixby, Capsules, Marketplace | Out-Domain Utterance Identification

Team

1. College Professor(s): Dr. Rajarajeswari S & Dr. K. Indira
2. Students:
    1. R Jeevan Kumar
    2. Angel Paul
    3. Ananya Muralidhar
    4. Shreyas Acharya
    5. Harsh Dutta Tewari
3. Department: Computer Science & Telecommunication

Date: 25th April 2022

Work-let Area – Bixby, Capsules, Marketplace | Out-Domain Utterance Identification

SAMSUNG

Problem Statement

- One significant problem with the classifier is identification of out-of-domain utterances - utterances which doesn't belong to any of the supported capsules.
- Creating a model capable of identifying user's utterance as out-of-domain so that Bixby can take appropriate action for it.
- To train out of domain utterances we need a rich/big data set of out of domain utterances which is difficult to get.
- The in domain utterances are easy to get from the training data present in our capsules. Using the in domain utterances as the data set we will build an AI/ML model which can detect out of domain utterances with better accuracy – one class classification problem

Sathwick Mahadeva,
Senior Technical Manager
sathwick.m@samsung.com
+91- 9886225372

Additional Documentation:

- https://www.researchgate.net/publication/221207574_A_Survey_of_Recent_Trends_in_One_Class_Classification
- https://www.researchgate.net/publication/312354057_Neural_Sentence_Embedding_using_Only_In-domain_Sentences_for_Out-of-domain_Sentence_Detection_in_Dialog_Systems

Expectations

Work-let expected duration – 4 months

3 Students

- Data set preparation/collection suitable for the task.
- Developing ML/DL model for identifying out of domain utterances.
- Understanding the implementation of existing solutions to identify their drawbacks
- Propose a method to solve the problem of identifying out of domain utterances which will solve the limitations of the existing solutions.

Training/ Pre-requisites

- Knowledge of NLP and Machine learning concepts.
- Hands on in Deep learning development concepts.
- Model development, training and inference on CPU and GPU

Kick Off < 1st Month >

- Understanding text processing and NLP concepts.
- Studying about one class classification problems.
- Understand the importance of one class classification in real world problems.
- Getting comfortable with ML and data pre-processing techniques

Milestone 1 < 2nd Month >

- Explore the state of art solutions available to resolve the one class classification problem
- Identify the drawbacks of the existing solutions.
- Data set preparation/extraction to work on the problem

Milestone 3 < 3rd Month >

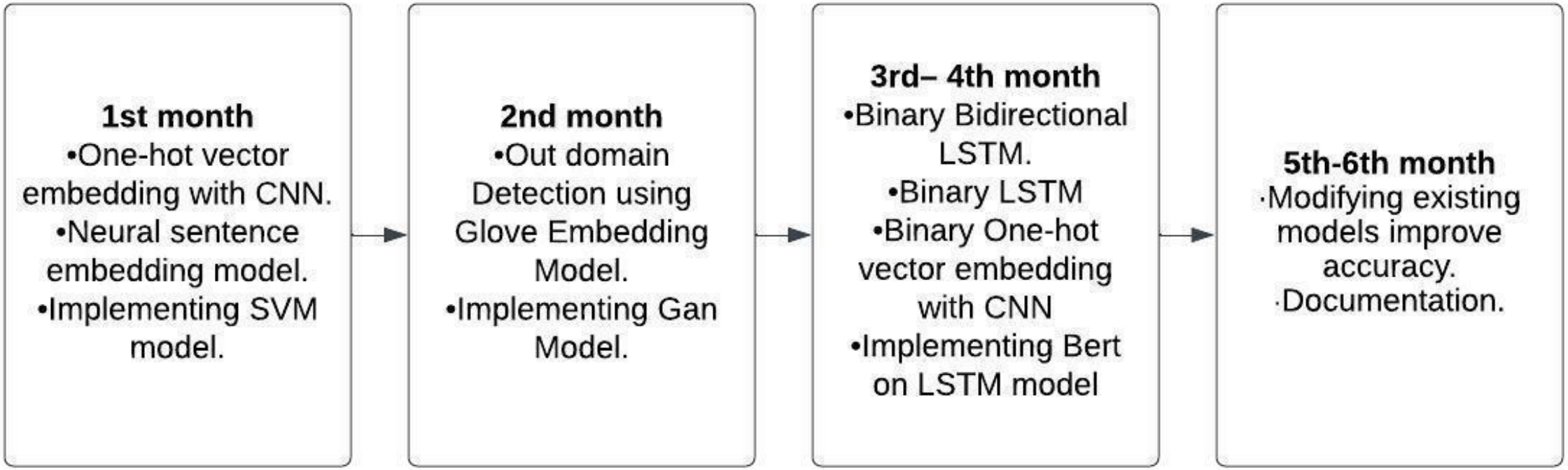
- Literature study to explore the new approaches to solve the problem of identifying out of domain utterances
- Propose and Build AI/ML model which will overcome the drawbacks of the existing solutions
- Enhance the model performance by tuning the hyper parameters.

Closure < 4th Month >

- Compare the performance of the proposed AI/ML model with existing solutions
- Evaluate if the proposed model can be integrated with bixby.

Approach / Solution

- **Concept Diagram** :
  (Clear detailed schematic / block diagram / flow chart depicting the proposed concept / solution)



Dataset(s) Analysis / Description

- **Dataset Capture / Preparation / Generation** :
  (Discuss the dataset generation process or if downloaded data provide details of what data & from where it was obtained etc… - 2 to 3 bullets only)

  - Source of first dataset:
    https://github.com/google-research-datasets/Taskmaster
  - Source of second dataset
    https://www.kaggle.com/stefanlarson/outofscope-intent-classification-dataset

- **Dataset Understanding / Analysis** :
  (Provide 2 to 3 bullets about what is your understanding of the data / opinion about the data)

The first dataset consists of 17,289 dialogs in the seven domains namely restaurants (3276), food ordering (1050), movies (3047), hotels (2355), flights (2481), music (1602), sports (3478)

The second dataset offers a way to evaluate intent classification models on ”out-of-scope” inputs. “out-of-scope” inputs are those that do not belong to the set of “in-scope” target labels.

- **Dataset Pre-Processing / Related Challenges (if any)** :
  (List out the challenges you fore see in data handling wrt problem definition – 2 to 3 bullets only)

  - Issues were caused by first dataset due to the presence of same utterances in both in-domain and out-domain datasets.

Dataset(s) Analysis / Description

- **Dataset Capture / Preparation / Generation** :
  (Discuss the dataset generation process or if downloaded data provide details of what data & from where it was obtained etc… - 2 to 3 bullets only)

Old Dataset:

Indomain : 12000
Outdomain : 2000
Training 9600
Test 4400

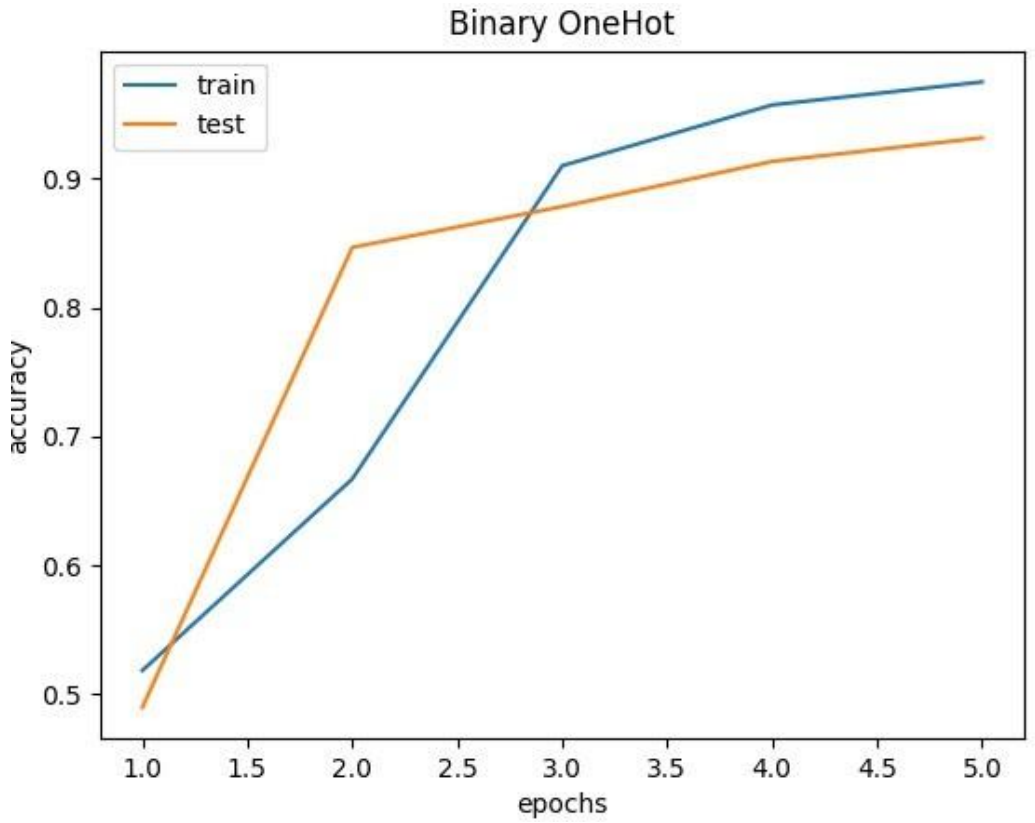
New Dataset:

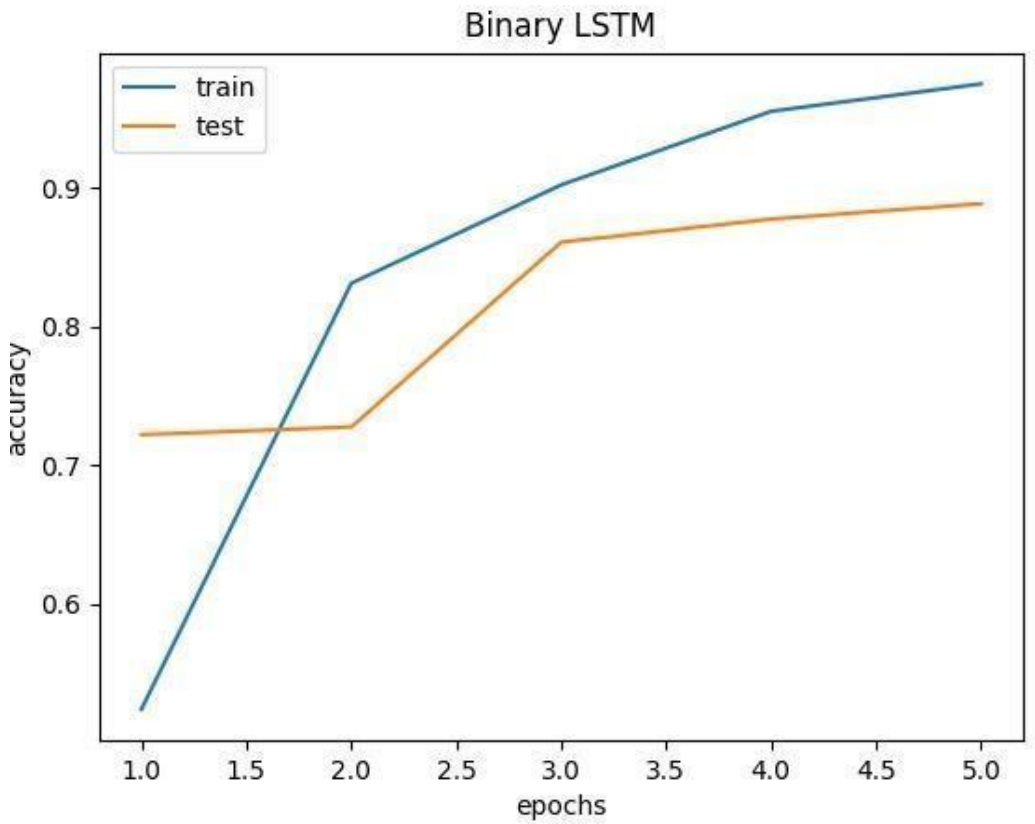
Is_train - 15000
Is_val – 3000
Is_test – 4500
Indomain: 22500

Oos_train – 100
Oos_val – 100
Oos_test – 1000
Outdomain: 1200

Experimental Results / Simulations / Observations

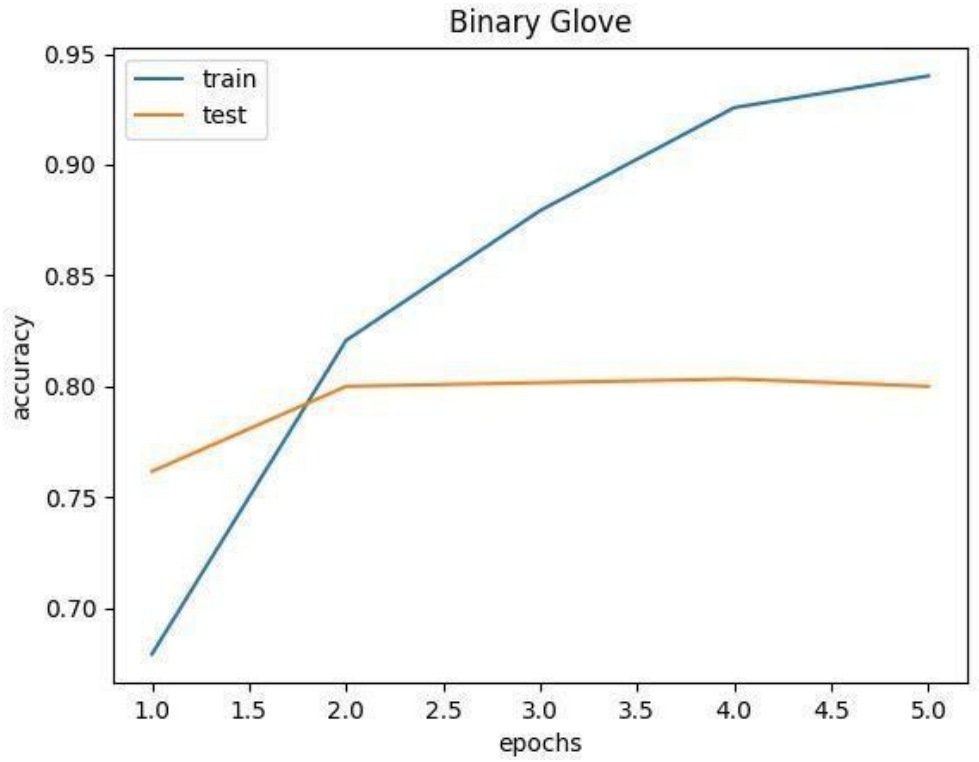
- **Results** :
  (provide numerical data / bar charts / plots / images / videos / tabulated results etc. Use full slide or multiple slides up to max 3 slides to demonstrate the results)

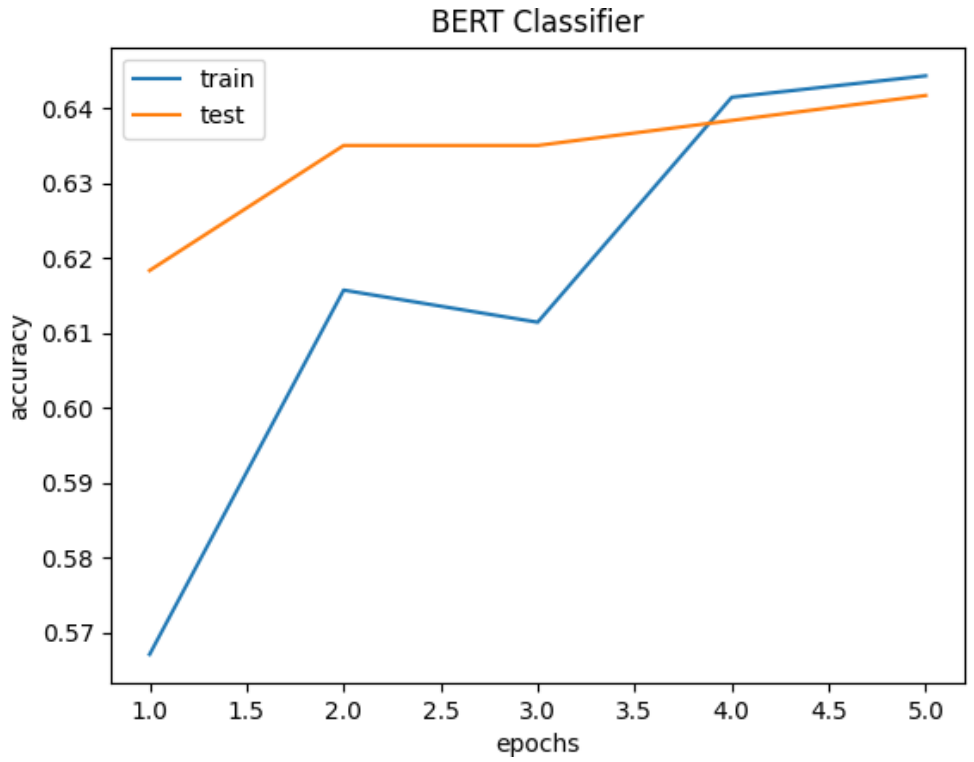




Experimental Results / Simulations / Observations

- **Results** :
  (provide numerical data / bar charts / plots / images / videos / tabulated results etc. Use full slide or multiple slides up to max 3 slides to demonstrate the results)

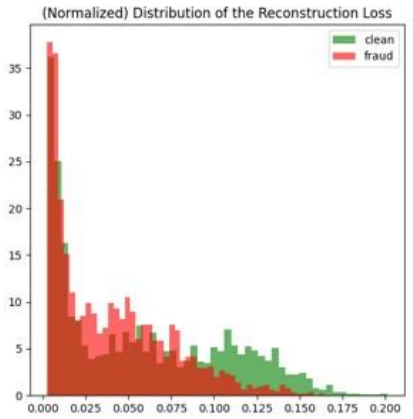
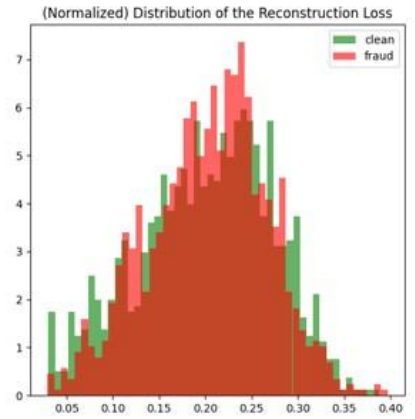
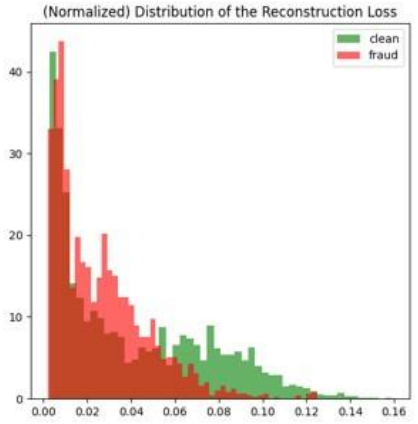
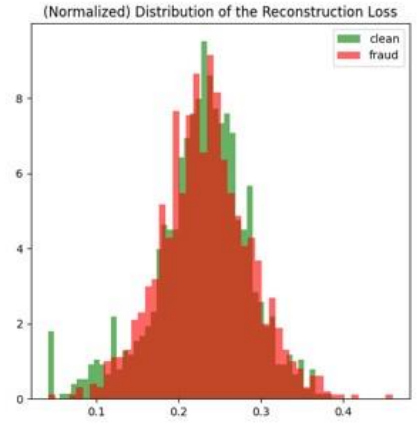




Experimental Results / Simulations / Observations

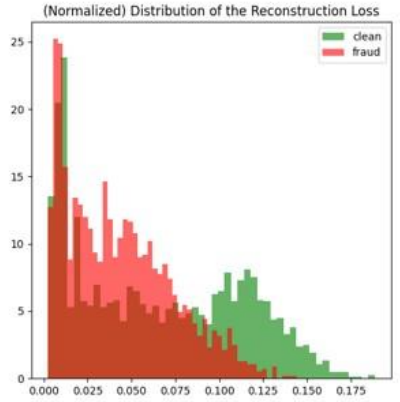
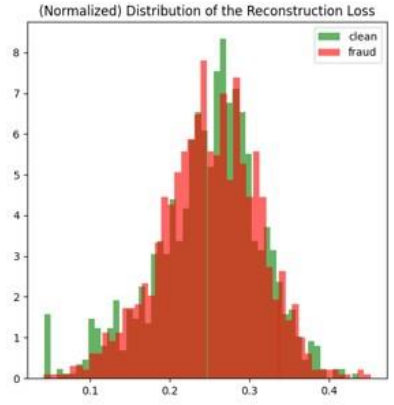
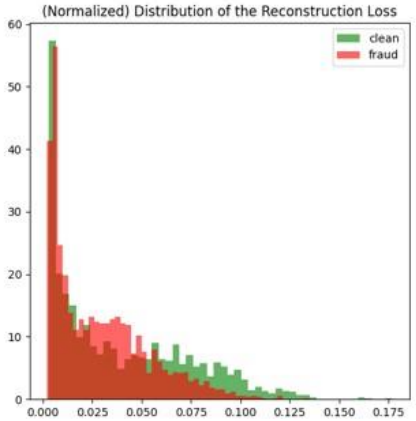
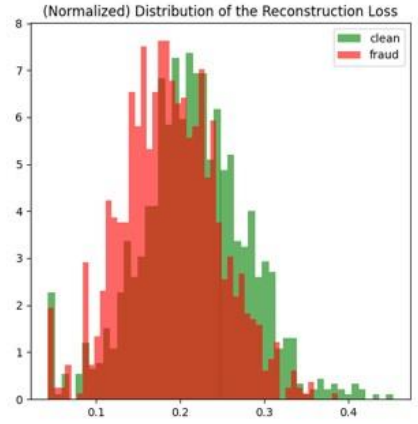
- **Results** :
  (provide numerical data / bar charts / plots / images / videos / tabulated results etc. Use full slide or multiple slides up to max 3 slides to demonstrate the results)

LSTM - Autoencoder

			Old Dataset			New Dataset		
Embedding Used	Embedding Dimension	Embedding name	Accuracy	Threshold	Reconstruction Loss Distribution	Accuracy	Threshold	Reconstruction Loss Distribution
Word2Vec	300	GoogleNews	0.5779	0.0807		0.4933	0.1988	
Glove	100	glove.6B.100d.txt	0.5945	0..525		0.4989	1999	

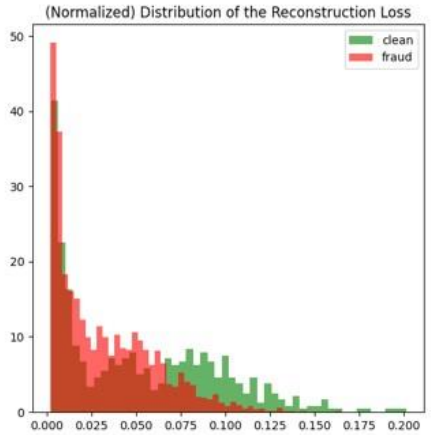
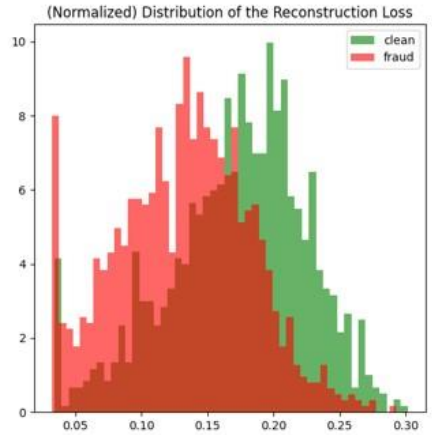
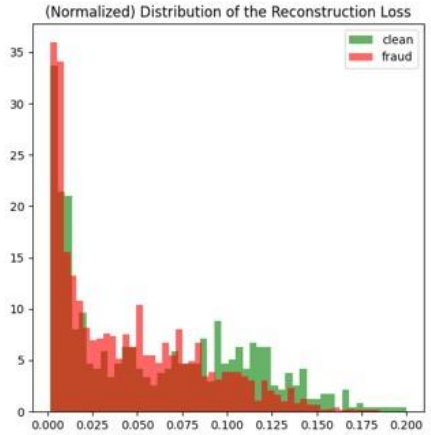
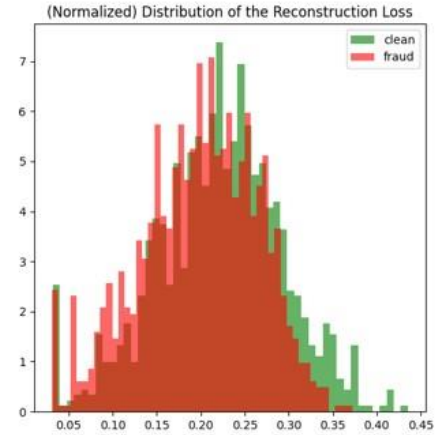
Experimental Results / Simulations / Observations

- **Results** :
  (provide numerical data / bar charts / plots / images / videos / tabulated results etc. Use full slide or multiple slides up to max 3 slides to demonstrate the results)

Glove	200	glove.6B.200d.txt	0.6168	0.0728		0.4881	0.1987	
Word2Vec	100	custom	0.5672	0.0513		0.5956	0.1958	

Experimental Results / Simulations / Observations

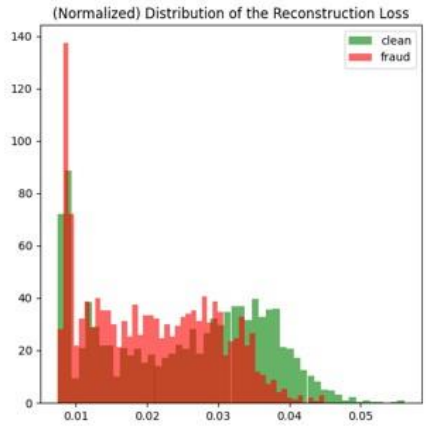
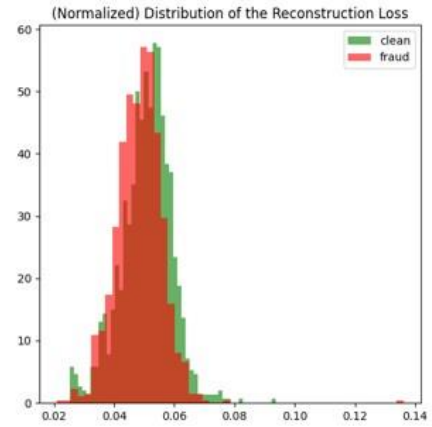
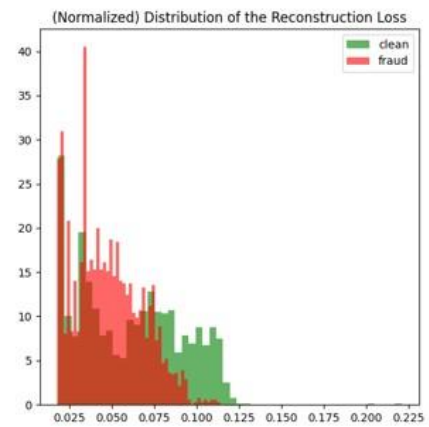
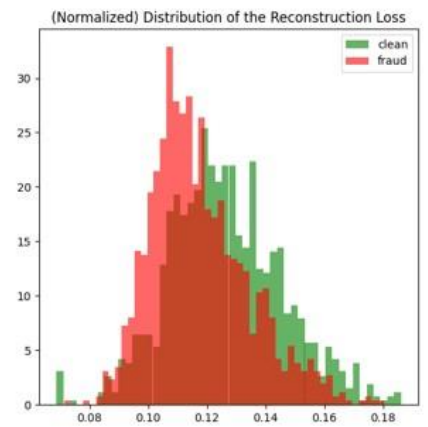
- **Results** :
  (provide numerical data / bar charts / plots / images / videos / tabulated results etc. Use full slide or multiple slides up to max 3 slides to demonstrate the results)

Word2Vec	200	custom	0.7915	0.0854		0.6748	0.1725	
Word2Vec	300	custom	0.73	0.923		0.5716	0.2165	

Experimental Results / Simulations / Observations

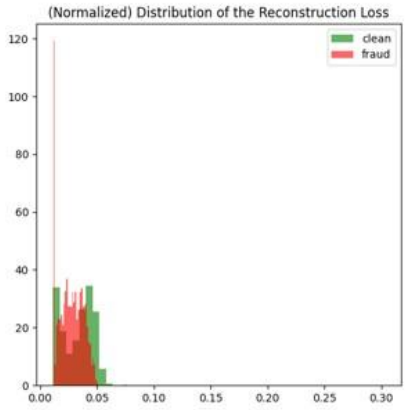
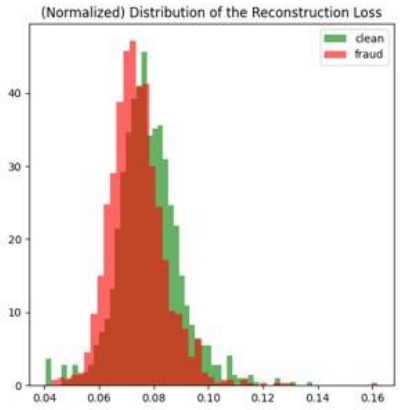
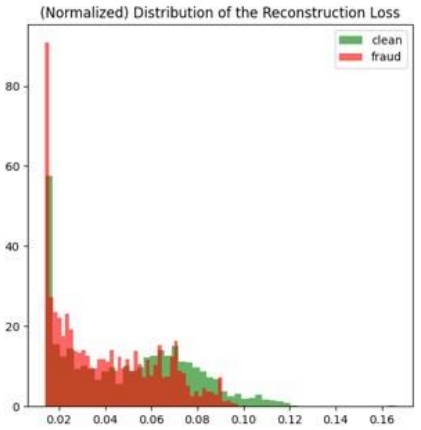
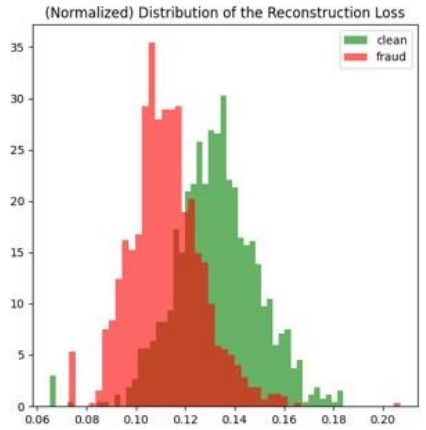
- **Results** :
  (provide numerical data / bar charts / plots / images / videos / tabulated results etc. Use full slide or multiple slides up to max 3 slides to demonstrate the results)

GAN

			Old Dataset			New Dataset		
Embedding Used	Embedding Dimension	Embedding name	Accuracy	Threshold	Reconstruction Loss Distribution	Accuracy	Threshold	Reconstruction Loss Distribution
Word2Vec	300	GoogleNews	0.5981	0.0285		0.5888	0.0537	
Glove	100	glove.6B.100d.txt	0.6143	0.0626		0.621	0.1185	

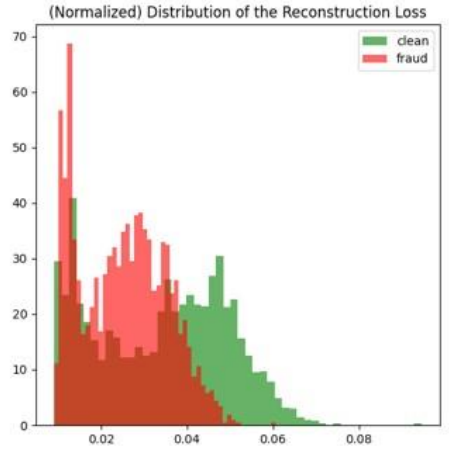
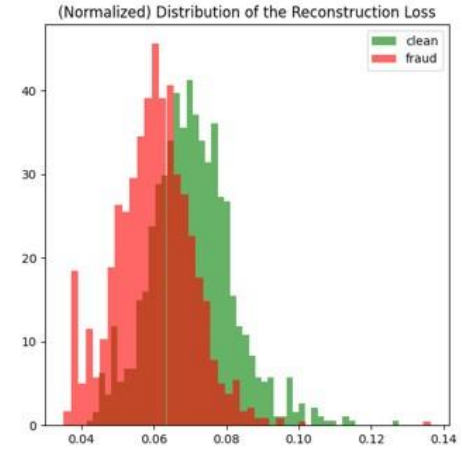
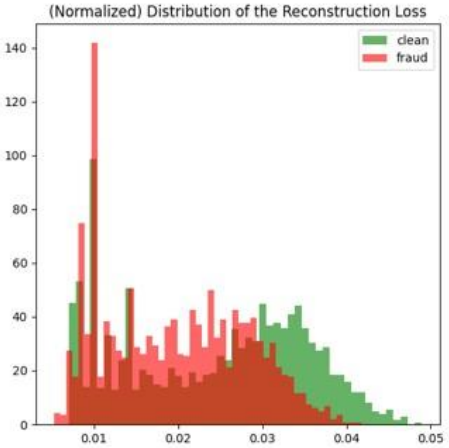
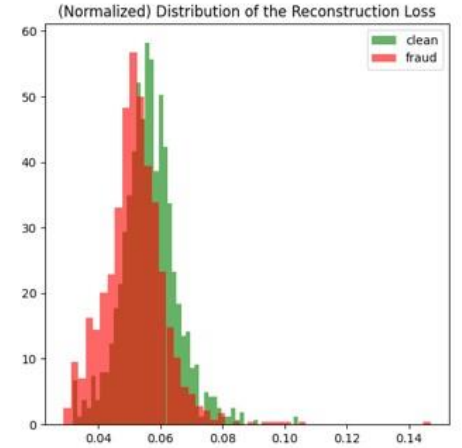
Experimental Results / Simulations / Observations

- **Results** :
  (provide numerical data / bar charts / plots / images / videos / tabulated results etc. Use full slide or multiple slides up to max 3 slides to demonstrate the results)

Glove	200	glove.6B.200d.txt	0.6304	0.0364		0.5965	0.0796	
Word2Vec	100	custom	0.5763	0.0481		0.7647	0.1195	

Experimental Results / Simulations / Observations

- **Results** :
  (provide numerical data / bar charts / plots / images / videos / tabulated results etc. Use full slide or multiple slides up to max 3 slides to demonstrate the results)

Word2Vec	200	custom	0.6536	0.0359		0.6954	0.0656	
Word2Vec	300	custom	0.6068	0.0257		0.623	0.55	

Experimental Results / Simulations / Observations

- **Results** :
  (provide numerical data / bar charts / plots / images / videos / tabulated results etc. Use full slide or multiple slides up to max 3 slides to demonstrate the results)

Model Accuracy Chart				
Classifcation	Model	Embedding Layer	Activation function	Accuracy (%)
Multi Class	Bidirectional LSTM	One Hot	Softmax	74.20%
Multi Class	LSTM	Keras	Softmax	72.80%
Multi Class	CNN	Glove	Softmax	70.10%
Binary	Bidirectional LSTM	One Hot	Sigmoid	93.40%
Binary	LSTM	Keras	Sigmoid	90.00%
Binary	CNN	Glove	Sigmoid	80.60%
Binary	LSTM-Autoencoder	Glove/Word2Vec	Sigmoid	79.15%
Binary	Bidirectional LSTM	BERT	Sigmoid	63.40%
Binary	One Class SVM	Count Vectorizer	Sigmoid	34.33%
Binary	Local Outlier Factor	Count Vectorizer	Sigmoid	49.00%
Binary	Isolation Forest	Count Vectorizer	Sigmoid	49.67%
Binary	GAN	Glove/Word2Vec	Sigmoid	65.35%

Deliverable

- **Final Deliverables** :
  (Discuss in the form of bullets, ~~what are the next steps to complete the solution~~, any road blocks / bottlenecks~~, any support needed from SRIB~~)

LSTM-AutoEncoder:
- Rise of issues due to the presence of same utterances in both in-domain and out-domain datasets.

Glove embedding model:
- Inability of Glove model to support context sensitive embedding.

GAN model:
- Problem in back propagating cross entropy loss.

Binary classification model:
- As the in-domain utterances increases 1:1 ration between in-domain and out-domain is hard to maintain.

- **IP / Paper Publication Plan** :
  (Details of papers / patentable ideas / innovative aspects that can lead to patentable ideas)

- **KPIs delivered/Expectations Met**:
  (Planned Expectations shared in Work-let vs Delivered Results)

Implementing multiple multi classification models and binary classification models and comparison of their accuracies, impact of each type of model and contribution of datasets in building better models.

Work-let Closure Details

- **Code Upload details:**

Items	Details
KLOC (Number OF Lines of codes in 000's)	2.529
Model and Algorithm details	LSTM, Autoencoder, SVM, BERT, Glove, OneHot, etc.
Is Mid review, end review report uploaded on Git ?	Yes
Link for Git	https://github.ecodesamsung.com/SRIB-PRISM/Bixby-Out-Domain-Detection

- **Data details (if applicable):**

Items	Data folder 1	Data folder 2
Name & Type of Data (Audio/Image/Video)	Utterances (.json)	Utterances (.json)
Number of data points	14000	23700
Source of Data (self collected, Scrapped, available on open source)	available on open source	available on open source
Google drive link/ git link to access data	https://github.ecodesamsung.com/SRIB-PRISM/Bixby-Out-Domain-Detection/tree/master/datasets/old	https://github.ecodesamsung.com/SRIB-PRISM/Bixby-Out-Domain-Detection/tree/master/datasets/new_kaggle

Note: If data uploaded on google drive, access to be shared to prism.srib@gmail.com

Thank you