

**R.V. COLLEGE OF ENGINEERING
BENGALURU – 560059**
(Autonomous Institution Affiliated to VTU, Belagavi)



**“Contextual Domain classification in spoken language
systems”**

PROJECT REPORT
Submitted by

SAUMYA

1RV13CS144

Under the Guidance of

Prof. Sharadadevi Kaganurmath
Assistant Professor,
Department of CSE, R.V.C.E.,
Bengaluru - 560059

Sagar Bhokre,
Senior Systems Software Engineer,
NVIDIA PVT.LTD.
Bengaluru - 560045

*In partial fulfilment for the award of degree
of*

Bachelor of Engineering
In
COMPUTER SCIENCE AND ENGINEERING
2016-2017

R.V. COLLEGE OF ENGINEERING, BENGALURU - 560059
(Autonomous Institution Affiliated to VTU, Belagavi)

Department of Computer Science and Engineering



CERTIFICATE

Certified that the Major Project work titled “**Contextual Domain classification in spoken language systems**” has been carried out by **Saumya (1RV13CS144)**, who is the bonafide students of R.V. College of Engineering, Bengaluru in partial fulfilment for the award of degree of **Bachelor of Engineering in Computer Science and Engineering** of the **Visvesvaraya Technological University, Belagavi** during the year **2016-2017**. It is certified that all corrections/suggestions indicated for the internal Assessment have been incorporated in the report deposited in the departmental library. The Major Project report has been approved as it satisfies the academic requirements in respect of project work prescribed by the institution for the said degree.

Prof. Sharadadevi Kaganurmath
Assistant Professor,
Department of CSE,
R.V.C.E., Bengaluru –59

Dr. G. Shobha
Head of Department,
Department of CSE,
R.V.C.E., Bengaluru –59

Dr. Subramanya K. N.
Principal,
R.V.C.E., Bengaluru-59

Name of the Examiners

External Viva

Signature with Date

1. _____

2. _____

1. _____

2. _____

R.V. COLLEGE OF ENGINEERING, BENGALURU - 560059
(Autonomous Institution Affiliated to VTU)

**DEPARTMENT OF COMPUTER SCIENCE AND
ENGINEERING**

DECLARATION

I, **Saumya (1RV13CS144)**, student of 8th Semester B.E., Computer Science and Engineering, R.V. College of Engineering, Bengaluru hereby declare that the Major Project titled “**contextual domain classification in spoken language understanding systems**” has been carried out by me and submitted in partial fulfilment for the award of degree of **Bachelor of Engineering in Computer Science and Engineering** of **Visvesvaraya Technological University, Belagavi** during the academic year **2016 - 2017**. We do declare that matter embodied in this report has not been submitted to any other University or institution for the award of any other degree or diploma.

Place: Bengaluru
Date:

Signature

Saumya

Acknowledgement

Any achievement, be it scholastic or otherwise does not depend solely on the individual efforts but on the guidance, encouragement and cooperation of intellectuals, elders and friends. A number of personalities, in their own capacities have helped us in carrying out this project work. I would like to take this opportunity to thank them all.

I deeply express our sincere gratitude to our guide **Prof. Sharadadevi Kaganurmath, Assistant Professor**, Department of CSE, R.V.C.E, Bengaluru, for her able guidance, regular source of encouragement and assistance throughout this project.

I would like to thank **Dr. G Shobha**, Head of Department, Computer Science & Engineering, R.V.C.E, Bengaluru, for her valuable suggestions and expert advice.

I also extend my cordial thank to **NVIDIA PVT. LTD.** for providing me an opportunity to carry out the internship in its organization. I also would like to thank my manager and all team members for their support and guidance.

I would like to thank **Dr. K N Subramanya**, Principal, R.V.C.E, Bengaluru, for his moral support towards completing my project work.

I thank my parents, and all the Faculty members of Department of Computer Science & Engineering for their constant support and encouragement.

Last, but not the least, I would like to thank my peers and friends who provided us with valuable suggestions to improve my project.

Abstract

Natural Language Processing is the application of computational techniques to the analysis and synthesis of natural language and speech and Machine Learning studies the generation and understanding of language, both in writing and speaking. Its main objective is to create machines able to understand and communicate in a natural way. Thus, chatbot are machines which can communicate with human. The global chatbot market has reached US\$ 994.5 million. Chatbot market for the period 2015 – 2024, has been forecasted to hit the market in US, Africa, India and other countries with huge IT sector. The market value importance increases as the development of conversational agent are in trend for the beneficial of the company and the easy access of the data for handling customers as well as providing easy access of the information. In a multi-area, multi-turn talked dialect understanding session, data from the history frequently significantly diminishes the uncertainty of the present turn. Illustrates the trained Neural system to misuse relevant data for inquiry area order.

The Convolution Neural Network classifies the spoken language statements into different Domains. This Convolutional Neural Networks for Sentence Classification accomplishes great arrangement execution over a scope of content grouping assignments (like Sentiment Analysis) and has since turned into a standard benchmark for new text classification structures. The model segregates the commands given by the co-pilot in car or automobile to perform particular task. Hence the commands are assigned to particular domain to which it belongs and then further actions can be taken to provide information to pilot. In this project, due to less availability of data it is restricted have to only three domains i.e., News, Control and Weather and trained them on CNN network to classify into three Domains.

The evaluation on this approach against SVM with and without contextual features. Contextually labelled dataset, observed a 1.4% absolute (8.3% relative) improvement in classification error rate over the non-contextual SVM, and 0.9% absolute (5.5% relative) improvement over the contextual SVM. Hence, this model can be used to deploy in any car system for further processing of text.

Table of Contents

Acknowledgement	i
Abstract	ii
Table of Contents	iii
List of Figures	v
List of Tables	vi
Chapter 1	1
Introduction	1
1.1 State of art development	2
1.2 Motivation	7
1.3 Problem Statement	7
1.4 Objectives	7
1.5 Scope	8
1.6 Methodology	8
1.7 Organization of the Report	8
1.8 Summary	9
Chapter 2	10
Text classification Techniques	10
2.1 Introduction to Text classification	10
2.2 Classification models	11
2.3 Prospect	13
2.4 Summary	14
Chapter 3	15
Software Requirements Specification of Classification	15
3.1 Overall Description	15
3.2 Specific Requirements	17
3.3 Summary	19
Chapter 4	20
High Level Design of Domain Classification	20
4.1 Design Considerations	20
4.2 Architectural Strategies	21
4.3 System Architecture	23

4.4 Data Flow Diagrams	24
4.5 Summary	27
Chapter 5	28
Detailed Design of Domain classification	28
5.1 Structured Chart	28
5.2 Functional Description of Modules	29
5.3 Summary	32
Chapter 6	33
Implementation Details of Domain Classification	33
6.1 Programming Language Selection	33
6.2 Platform Selection	33
6.3 Coding conventions	35
6.4 Difficulties Encountered and Strategies used to tackle	38
6.5 Summary	38
Chapter 7	39
Software Testing	39
7.1 Test Environment	39
7.2 Unit Testing	39
7.3 Integration Testing	45
7.4 System Testing	46
7.5 Summary	46
Chapter 8	47
Experimental Results and Analysis for Classification Network	47
8.1 Performance analysis of results	47
8.2 Summary	49
Chapter 9	50
Conclusion and Future Enhancement	50
9.1 Limitations of the Project	50
9.2 Future Enhancements	51
References	52
Appendices	57

List of Figures

Fig No.	Figure Name	Page No.
Figure 4.1	System Architecture for Chatbot	24
Figure 4.2	Level-0 Data Flow Diagram	25
Figure 4.3	Level-1 Data Flow Diagram	26
Figure 4.4	Level-2 Data Flow Diagram	26
Figure 5.1	Structural Chart of Domain Classification	29
Figure 5.2	Learning Module Flowchart	30
Figure 5.3	Network Flowchart of word embeddings and output	31
Figure 6.1	TensorBoard graph of Network	35
Figure 8.1	Graph showing loss	47
Figure 8.2	Graph showing Accuracy	48
Figure A.1	Dataset of three domains	57
Figure A.2	Training Process	58
Figure A.3	Evaluation screenshot	58
Figure B.1	Domain Classification Web UI	59
Figure B.2	Output of the Classification	60
Figure B.3	Accuracy graph through Tensorboard	60

List of Tables

Table No.	Table Name	Page No.
Table 7.1	Input Processing Test	40
Table 7.2	Testing Sentence of news Domain	40
Table 7.3	Neural network construction	41
Table 7.4	Testing Sentence of Control domain	41
Table 7.5	Unit test for Training process of the network	42
Table 7.6	Testing Sentence of unknown Domain	42
Table 7.7	Testing Sentence of weather domain	43
Table 7.8	Testing Dataset with other ascii input	43
Table 7.9	Testing Different domain sentence	44
Table 7.10	Testing for Words not in vocab	44
Table 7.11	Testing Sentence of news Domain	44
Table 7.12	Integration testing for Domain Classification	45
Table 7.13	System Testing	46

Chapter 1

Introduction

Spoken Language Understanding applications are getting to be progressively critical in our day by day lives. Numerous convenient gadgets, for example, cell phones have individual aides that are worked with Spoken Language Understanding advancements. Spoken Language Understanding normally includes deciding the client plan and separating important semantic spaces from the normal dialect sentence. In a regularly utilized design, a sentence is initially characterized into one of the upheld areas, after that space subordinate expectation examination and opening filling (i.e. element extraction) are completed [1].

In such pipelines, space arrangement is the initial step of the semantic examination. While it is a standard arrangement assignment what's more, apparently less intricate than other semantic examination such as element extraction, the blunders made by a space classifier are typically substantially more obvious – they frequently prompt plainly wrong framework reactions, for example, conjuring a wrong application since it courses the question to wrong purpose and space models for semantic investigation. The concentration of this work is to enhance the exactness of space arrangement by abusing the session setting [2].

Normal Language Processing is the concentration of computerized reasoning exploration and has numerous applications: machine interpretation, named entity recognition, address replying, and so forth. The reason for a QA framework is to consequently answer questions postured by people in a characteristic language. Knowledge–Base upheld (KB–supported) QA framework [2] acquires replies by questioning from an organized information database, which stores tuples of (entity, connection, esteem), and producing answers in regular languages as indicated by the inquiry result. For instance, for the question "Who's the leader of the Assembled States?" the entity is "USA", the connection is "be–president–of “, and the esteem is "Barack Obama". Understanding human inquiries, particularly separating the entity and connection applicants, is the first and key stride toward executing the entire framework. Numerous conventional techniques rely on upon catchphrases or layouts coordinating. In any case, they depend intensely on hand–crafted rules, which can't be scaled up. To use human work

in building the catchphrases or layouts, some current machine learning calculations have been proposed to consequently learn elements and measure semantic separations amongst inquiries and known areas [3].

1.1 State of art development

In this section, it has been explained the challenges related to contextual domain classification in spoken language system.

1.1.1 Need for a generalized Learning agent

The attempt to make a generalized reinforcement learning has been in under process for a long time. The recent use of GPUs for neural networks and new frameworks for machine learning like Tensorflow and Theano makes it feasible and convenient build and train neural network in short time. As this project uses screenshots from the environment as input, it takes a long time for each epoch. This seamless integration with Tensorflow has made it possible to test multiple network configuration. Tensorboard, a web interface from Tensorflow helps to visualize the graphs in training process and analyze the loss and other weights in real time during training [4].

A Lot of researchers across institutions have discovered many ground-breaking solutions for deep neural network and machine learning that can help project.

Mioa Sun [5] proposed a changed rendition of mSDA algorithm (mSDA++). Likewise, considering that words in various areas may express extraordinary opinion, joined mSDA with EASYADAPT algorithm (EA+mSDA). At that point have connected SVM (baseline), mSDA, mSDA++ and EA+mSDA algorithms on cross-area feeling arrangement of item audits. The trial comes about demonstrate to us that EA+mSDA algorithm accomplishes the best exactness of arrangement. Also, mSDA++ algorithm can quicken the consequent estimation and diminish the information storage room yet, likewise saves solid element learning capacities. From the outcomes, they felt that profound learning has exceptionally solid abilities to learn information portrayals.

Alfan Farizki Wicaksono [6] demonstrated that, with just straightforward pack of words components, RNNs beats customary machine learning models for basic sentence characterization issue. This outcome was very line with the way that RNNs can use data from coterminous positions, which is something that other customary models can't normally do. However, a few addresses still stays unanswered. For instance, what will happen when utilized better and more discriminative components as well as perform adequate component building? Will RNNs still outflank conventional machine learning models? On the off chance that the appropriate response is "yes" will it be sufficiently commendable to supplant different models with RNNs? The later question could be replied by exercising a few viewpoints, similar to importance of exactness change and preparing time intricacy.

ZHENG-TAO YU [7] through choosing feature space, using domain knowledge to adjust the feature weights of domain text, using SVM algorithm, the paper constructs a domain text classification model, and an experiment was done based on Yunnan tourism domain. The results prove that domain knowledge relations have a good influence on the domain text classification. The domain text classification accuracy rate has been improved. Further research will aim on the relevance between domain term, the specific impact on text classification of the relevance between domain terms and multi-domain classification model.

Guofeng Zhu [8] propose the Domain Classification Calculation Based on KNN in Micro-blog. This calculation full considers the normal for intersectional and interdisciplinarity about field in this paper. In conclusion, this technique can unmistakably appraise the impact about clients on the whole strolls of life. It is anything but difficult to get the most recent element about all fields also, understand the insightful suggest work about each field successfully as well. Furthermore, it is important to cognize client's specialty and decide the key system showcasing. Unquestionably this paper additionally has the accompanying restrictions: the productivity of Arranging isn't sufficient high and don't give enough weight to the field that clients occupied with. Later on, work, concentrated on enhancing the productivity of the field grouping and full considering the heaviness of field that the client occupied with.

Inseok Heo [9] This paper demonstrates a strategy for separating relative planning data from the discourse of numerous speakers saying a similar thing. A technique for transitive approval is acquainted with check the conduct of the strategy, and the yield timing maps can be utilized as elements in a characterization conspire. The precision of arrangement utilizing the planning maps is contrasted and grouping in view of ghastly techniques in two examinations that endeavor to consequently mark a speaker's sexual orientation and whether they are a local speaker of English.

Alexander C. Nwala and Michael L. Nelson [9] characterize an arrangement of elements in SERPs that show if the space of a question is academic or not. Our classifier which has an exactness of 0.809 and F-measure of 0.805 can be further connected to different spaces once discriminative and instructive elements are recognized.

Neethu Akkarapatty and Nisha S Raj [10] Distinctive style markers are utilized on a benchmark dataset named TripAdvisor. For the most part components 2016 third International Conference on Signal Processing and Integrated Networks (SPIN)320 from four classes to be specific Character, PoS tag, Bag of Words what's more, Aspects were separated. It has been seen that execution is enhanced when prevent words are expelled from content before model development. Dimensionality decrease was connected in this manner utilizing Pearson's Chi-Square all together to pick the noticeable elements from a colossal element space. A vector space show for pertinent elements was developed and given to the classifier utilizing WEKA instrument.

Francesc Alías [11] has presented our proposition towards enhancing the adaptability of high caliber TTS frameworks by considering numerous spaces, named multidomain TTS amalgamation. This proposition has a place with a current research heading concentrated on joining further content examination for taking TTS frameworks somewhat nearer to human conduct—which frequently incorporates diversion changes, distinctive talking styles, and so on., inside a similar discussion.

B.U. Anu Barathi [12] In this paper it compares the classification results of the Semantic approach based on the BOW representation of documents, and based on the extended Distance Based Model. The Semantic algorithm uses a classifier to initialize the out-of-domain documents into clusters. Thus, they also reported the results, with and without enrichment, respectively. Standard pre-processing was performed on the raw data. All letters in the text were converted to lower case, stop words were eliminated, and stemming was performed using the Porter algorithm. Words that appeared in less than three documents were eliminated from consideration. Term Frequency was used for feature weighting when training the frequent document, and for the Semantic algorithm.

R. Makineni [13] have proposed the learning advantage and contended how a machine in view of this concept, a information advantage machine could be worked to expand the efficiency of learning worker. The proposed multi specialist display known as Vijjana shows how distinctive elements of this machine could be dealt with by an accumulation of a commonly participating operators.

Saeedeh Momtazi and Dietrich Klakow [14] exhibited a dialect show based approach for characterizing sentences as opinionative and verifiable with regards to sentiment address replying. They utilized a Bayes classifier with various smoothing strategies and diverse ngram models. The outcomes demonstrate that our proposed approach essentially enhances the sentence arrangement execution furthermore, outflanks the SVM which is the best order strategy in the accessible writing.

Siddharth Pal, Yuxiao Dongy, Bishal Thapa, Nitesh V. Chawlay, Ananthram Swamiz, Ram Ramanathan [15] displayed three methodologies that investigate the viability of profound learning to network investigation. While still extremely preparatory, they accepted that our initial outcomes demonstrate that profound learning has the potential for portrayal learning in systems. Future bearings incorporate exploring different avenues regarding expanded number of layers, utilizing chart "themes" in the hunt convolution, also, preparing with a bigger corpus of information, including engineered and certifiable information.

Suman Ravuri and Andreas Stolcke [16] analyzed customary and neural word grouping classifiers on the assignment of area order, utilizing a vast true corpus (Cortana client input), while additionally considering troupes of double versus n-way order structures. The outcomes are to a great extent in line with earlier outcomes on double articulation order, with gated unit repetitive systems – LSTM and GRU – performing best, and beating (at any rate in general precision) the baseline frameworks.

Yangyang Shi, Yi-Cheng Pan, Mei-Yuh Hwang, Kaisheng Yao, Hu Chen, Yuanhang Zou, Baolin Peng [17] an imperative component in multi-lingual talked dialect understanding area arrangement, particularly when the preparation information is acquired by means of machine interpretation where the high request n-gram elements are blended. To display the second request co-event viably, they have proposed a strategy in light of factorization system. Contrasted and polynomial direct SVM that additionally models cooccurrence, the proposed strategy has significantly littler model size. The proposed technique accomplished comparative execution with polynomial linear SVM.

Hongjie Shi, Takashi Ushio, Mitsuru Endo, Katsuyoshi Yamagami and Noriaki Horii [18] In this work connected three distinctive installing models, while there is one more they didn't attempt - the English character show. There are a few character-mindful dialect models proposed as of late, which are prevalent in managing subword data, uncommon words and incorrect spelling. They trusted that coordinating them into the multichannel model is a promising examination course.

Rui Zhao and Kezhi Mao [19] have presented a theme mindful profound compositional system and created two models named TopCNN and TopLSTMs for sentence portrayal. By joining likelihood dispersion inferred by LDA, TopCNN and TopLSTMs can investigate the theme feeling of words and sentences and further use such subject particular sense mutually with general feeling of word embeddings in the semantic compositional process.

Lang Zhining, Gu Xiaozhuo, Zhou Quan and Xu Taizhong [20] have presented a theme mindful profound compositional system and created two models named TopCNN and TopLSTMs for sentence portrayal. By joining likelihood dispersion inferred by LDA, TopCNN and TopLSTMs can investigate the theme particular feeling of words and sentences and further use such subject particular sense mutually with general feeling of word embeddings in the semantic compositional process.

The above newly introduced papers proposed a new way to solve large-scale reinforcement learning challenges. These proposed methods are very helpful and provide a concrete insight into the conversational agents mind and help in many ways to improve this project. For the proposed project, used a CNN model with low learning rate and high epoch value so that learning process is slower with proper increments.

1.2 Motivation

Conversational agent is a next generation use of machine learning in artificial intelligence. This is especially useful for robotics, self-driving cars, autonomous vehicles, optimization control and many arrays of scientific research. This opens up the field for a completely new set of machine learning application that over time can improve the performance and contribute where humans lag in real-time. Hence the idea behind this project is to contribute to next generation artificial intelligence technology for bettering human-machine interaction.

1.3 Problem Statement

The Spoken Language System is difficult for the Computer to understand and tackle each of the users specially for the big companies and hence need an automated computer to interpret these languages and provide the information to the user without much man effort.

1.4 Objectives

The objectives of this project are to develop a distributed framework for reinforcement learning purpose that can generalize for any sentence. This is designed with the help of Tensorflow that provides a suitable machine learning framework for training a neural

network and modification. From the framework develop CNN which is trained on dataset generated by the company and can classify the commands into different domain for its further indent and entity recognition.

1.5 Scope

The system is designed for reinforcement learning on a GPU and distributed architecture. The framework is compatible with any environment. As CNN is made for an environment with a medium number of dynamic programming tree, the network has its limitation on its estimation of action value. The framework is programmed to work on Nvidia graphics card and can work on any large-scale machines i.e. up to 256 distributed machines. So, this can be used as an introductory Reinforcement Learning framework to train a large number of different applications on real-time.

1.6 Methodology

The agent learns the environment as states. Each state has a particular value and coming to that state has its own reward. This is learnt by the system as it moves around in the dynamic programming graph. But for all the environment, dynamic programming is hard to know for the agent. So the agent follows a process called Markov Chain model, where the dynamic programming tree is learned as the agent moves through the environment by taking action. The action value for each action is approximated by using special network named CNN [21]. The dropout from every action from the network are then fed to the network in the next phase as feature vector.

1.7 Organization of the Report

This section gives a broad picture of the various chapters in the report.

- **Chapter 2** is the overview of the CNN Learning describing the details of the domain of the project.
- **Chapter 3** is the Software Requirement Specification which describes the user characteristics, dependencies, constraints and the functional requirements for the project.
- **Chapter 4** is the High-Level Design which describes the design phase of Software Development Life Cycle. This chapter discusses about the design considerations

such as general constraints and development methods. This chapter also explains the project System Architecture and Data Flow Diagrams.

- **Chapter 5** is the Detailed Design which explains about the project modules. The functionalities of each module are explained in this section. Structural diagram of the system is also explained in this chapter.
- **Chapter 6** is the Implementation details which describes the technologies used in the system. This section also explains programming language, platforms, environment and the code conventions followed.
- **Chapter 7** is the Software Testing which explains the test environment and briefly explains the test cases executed during unit testing, integration testing and system testing.
- **Chapter 8** is the Experimental Results which mentions the results found during the experimental analysis of the available data. It also tells about the inference made from the results.
- **Chapter 9** is Conclusion conveying the summary, limitations and future enhancements of the project.

1.8 Summary

This chapter gave a brief introduction to the project with few basic concepts and explained the scope, methodology and organization of the report.

Chapter 2

Text classification Techniques

Text classification, also called topic classification, is a methodology to categorize pieces of text into classes. It has a very interesting business application that some companies are taking advantage of. When a customer communicates with the company asking for a service or just for customer support, the system has to redirect the customer to the correct department.

2.1 Introduction to Text classification

Text classification, additionally alluded to as record characterization, content order or archive classification is an undertaking to allocate the content records into some predefined classes.

The world has developed in many-sided quality, overpowering us with quickly developing measure of information. With the exponential development of the web, an extraordinary premium has been contributed towards creating proficient devices to successfully deal with the vast and complex information. Because of expanding measure of data that is put away in computerized or electronic frame, content reports are developing quickly. A noteworthy piece of this data is put away in content databases and archives which contain colossal arrangement of archives having a place with different sources like research papers, email messages, news articles, diversion, craftsmanship, writing, books, site pages and advanced libraries. In the vast majority of associations these days, data is kept as content databases like in wellbeing, government, instruction, business and different different zones. The content put away in the databases is in intrinsically unstructured or semi organized organization. The developing monstrosity what's more, many-sided quality of this unstructured content information has made it an essential need to oversee it adequately. This should be possible by grouping the records to streamline the capacity, recovery and introduction of content information. Manual information examination has progressed toward becoming exceptionally dull because of substantial dimensionality of content

information. In such a case, content mining is the main want to deal with this tremendous gathering of information [18].

2.2 Classification models

The model of the content classification is the centre some portion of the classification procedures; its primary target is to enhance the classification consequences of the review. The pragmatic models must likewise consider the capacity and registering power with limited conditions, adaptability and the classification procedure of the learning procedure throughput rate (speed). There are various classification models which commonly utilized: Boolean Model, Linear Model, Probabilities Model, INN Demonstrate, Nonlinear Model, Composition Model and Text Categorization Model Based on Semantics [19].

2.2.1 Boolean Model

Boolean Model is entirely in view of highlight coordinating display. To begin with, this model sets up an accumulation of parallel factors which relating to the qualities of the content. These qualities of factors speak to the content and if there is a comparing highlight, the element factors are assessed to "Genuine"; something else, the element factors are taking "False". Be that as it may precisely match of the Boolean Model can return an excessive number of or excessively few aftereffects of the content [20].

2.2.2 Linear Model

Utilization of SVM (Support Vector Machine) is one of the most imperative advances as of late; its rule is a hyper plane straight parcelling on the given preparing set; this will change over the arrangement issue into a spatial ideal plane issue, at the end of the day into a quadratic programming issue. In spite of the fact that SVM which is prepared on substantial informational index requires a lot of capacity assets and an abnormal state of processing power in, its mode isolated with great speculation adequately conquers the effect variables, for example, the over-fitting and the example dispersion excess elements. Reference isolates expansive informational collection into littler informational index to prepare the Support Vector Machines; and after that the model consolidations two of two outcomes by consolidating calculation; at last, it frames the last characterization model of Support Vector Machine [20].

2.2.3 Probabilities Model

The most ordinary algorithm is Naive Bayes algorithm in Probabilities Model. Despite the fact that characterization capacity of Bayes technique is weaker, this model is supported worried with changes and upgrades due to its straightforward, effective.

The primary thought of the algorithm depends on Bayesian suppositions, that is to state, vocabulary in the reports decide the content classification on the part of autonomous. To begin with, this technique computes the priori likelihood of trademark words which have a place with every classification; and when the new content arrives, it computes the trademark expressions of the content which have a place with every class of the back likelihood agreeing to priori likelihood; at last, it takes a most extreme posteriori likelihood class as the order comes about [21].

2.2.4 Non-Linear Model

The common calculation in Non-Linear Model is Decision Tree. The calculation develops a top-down tree-sort structure recursively, it makes a branch for each known estimation of test property from properties with most extreme data pick up to group tests which have famished testing all properties, for example, ID3, C45, C5 et cetera [21].

2.2.5 Combination Model

The possibility of Combination Model is to pick a few diverse grouping calculations; its point is to enhance the general execution of the characterization framework as indicated by the general investigation and judgment for the consequences of a few calculations. In content characterization, utilizing a few diverse classifiers to judge a content classification, and after that union the outcomes in a suitable way. Reference consolidates the Boosting calculation and Machine Learning calculation to enhance gauge exactness [21].

2.2.6 Text Classification Model based on Semantic

This model proselytes word record free of each other from high dimensional element spaces to low-dimensional semantic spaces or the idea of highlight spaces by utilizing the connections between words in metaphysics, conglomeration, mining, for example, idle semantic ordering semantic. So this can prepare the content arrangement demonstrate. Idle semantic ordering is an innovation of archive ordering and content arrangement, this

predominantly produced semantic vector show by co-event words, and basically takes care of the issue of high measurement of record vectors through word-content grid particular esteem deterioration procedure in [21].

2.2.7 INN Method

The method ascertains the comparability between test documentations and preparing records. The average calculation is K-Nearest Neighbour calculation. The primary thought of this calculation is to find out the nearest message in the preparation set and test set by computing, that is to state, to locate the few neighbours of this content and decide the classification of this new content as per the sort of its neighbours. Choose to test tests of the classification, that is condition to test the tests of the K-Nearest Neighbours, it does not take into account that the K-Nearest Neighbours in their particular classes at the level of significance. For this issue, Reference utilizes the grouping calculation to discover the preparation tests of each preparation test accumulation enrolment; they likewise utilize the enrolment to separate between test of the K-Nearest Neighbour [21].

2.3 Prospect

Text classification innovation has been exceptional to determine the examination of most information volume and littler callout, the information uniform dissemination, the issue of components and application. In any case, consequently message classification innovation of mass application is as yet numerous issues in present, and new classification strategies always rise [22].

2.3.1 Fusion of Feature Selection and Feature Reconstruction

Feature Selection Method expect that trademark measurements is autonomous in highlight determination; it doesn't take into record semantic relationship in the component measurements, which diminishes the precision of arrangement. Consequently, pattern of content classification is change and improvement of highlight choice, and highlight combination between recreation strategy, for example, highlight determination of combination common data and clustering [22].

2.3.2 Scalability of Algorithms

Substantial scale content arrangement has turned into a squeezing require because of monstrous data and complex literary substance on the Internet. Vast scale content arrangements are confronting with countless and the quantity of preparing tests, which brings both the calculation of the calculation time and capacity with the development of classification. Multiplayer classification is a decent approach to unravel sizes of calculations time. Along these lines, vast scale content characterization is an essential research field [23].

2.3.3 Semantic or Concept of Vector Space Model

As impedance of polysemy and equivalent words, semantic content order strategies or idea vector space show move toward becoming more well-known review territories for scientists in the field. As of now there are a ton of semantics or the models of idea of vector space, for example, dormant semantic examination, philosophy mapping strategy, the idea grid development technique and institutionalized strategy for idea examination et cetera. Reference proposes SVM content characterization technique in light of idea grid and institutionalization ideas to develop idea grid for the preparation of content grouping model through examining connection between mining archive qualities in 2009 [23].

2.3.4 Integration, Improvement of Text Categorization Algorithms

At present, there have been numerous substantial content arrangement calculations, which have points of interest and detriments [24]. Instructions to utilize these calculations viably to enhance the execution of content grouping calculations by utilizing their favourable circumstances to coordinate, enhance and increment. As per the utilization of content arrangement foundation, portrays the techniques for multi-order with highlight determination.

2.4 Summary

This chapter explained the basics of Text classification and its aspects. It also explained various Text classification models and categories are also explained.

Chapter 3

Software Requirements Specification of Text Classification

Software Requirements Specification is a detailed description of the whole system to be developed [25]. It includes Functional and Non-functional requirements of the system. Functional requirements of the system describe what the system should do. It focuses on what an outside agent observes while interacting with the system. It also includes the Purpose, Process and the Methods of the system. The Non-functional requirements are the constraints for the design and implementation of the system. These specify the criteria used to judge the operation of the system rather than any specific behaviours.

3.1 Overall Description

This section describes the general factors which affect the system and its requirements. The application developed should provide a user-friendly interface to run and observe the results conveniently. This section also deals with user characteristics, constraints on using the system and dependencies of the system on other applications.

3.1.1 Product Perspective

The system, as a product, needs to be versatile and easy to use. At the same time, it must be scalable to performance. It must be modularized properly in order to make the maintenance of the codebase easier. Lastly, it must be deployable to production [25].

The intended users of the system are mainly Machine Learning or Artificial Intelligence researchers, who are keen in developing new algorithms or improving CNN hyperparameters [25]. It can also be used by students to learn more about the implementation of algorithms in practice, observe how various parameters change over time, find patterns in the agent learning the environment and use their knowledge to improve the same. One can also improve our system to work efficiently across many systems, distributed training, efficient hardware utilization to decrease the training time.

3.1.2 Product Functions

The primary function of the system is to learn any kind of environment that is given in the form of word vector, although currently, focused on commands, as they represent simpler form and convenient to test algorithms [25]. The system is written using Tensorflow library released by Google Inc., which provides an interface called Tensorboard to observe all the variables changing in real-time. This makes debugging process easier and find mistakes, improvements on the fly. Also, designed a web interface to observe additional information such output, the reward achieved continuously over time. It makes easier for one to look at the algorithm performance directly and predict the total time required to learn the environment to a super-human level.

3.1.3 User Characteristics

The users of the system will have basic knowledge about Machine Learning, experience in designing machine learning systems in practice [25]. Particularly, one would have a basic understanding of Convolutional neural network and primary algorithms. Decent experience with Python programming language is a primary need for the users in order to work with our system which uses high-level APIs like Tensorflow written in Python.

3.1.4 Constraints and Dependencies

The system is dependent on the following factors:

- The amount of dataset fed to the network.
- A high-end system with sufficient memory and graphics card is a must to speed up the training process
- While training the agent, the system should be up and running without interference.
- In a distributed environment, network speed must be very high to share parameters between nodes.

3.2 Specific Requirements

This section explains about the Software Requirement Specifications (SRS) [25] to the level of sufficient details. While the product perspective and the User characteristics do not state actual requirements the system to be satisfied, the Specific requirements are the actual data the user needs to agree, in order to use the system.

3.2.1 Functional Requirements

Functional requirements [25] of the system are the following:

- The agent must be able to learn an arbitrary environment, given as raw text input
- The system must display all the variables to visualize in real-time
- Must provide an interactive interface for data visualization
- The program must utilize the computation unit available to maximum
- Results of training must be easily accessible and displayed.
- System must be able to resume the training process from checkpoints

3.2.2 Performance Requirements

Performance requirements of the system include the following,

- The program must use efficient Python data structures (such as *numpy* arrays) to reduce the memory consumption.
- GPU Utilization must be very high to maximize the performance and therefore reduce training time
- The program must utilize the available nodes in distributed environment for a maximum amount of time, so that none of them be idle.

3.2.3 Supportability

The program for the algorithms is written in Python programming language, thus it can be executed in any Platform where Python is installed. The Web Interface designed to visualize the results supports all the modern web browsers [26]. The program uses Tensorflow library, hence if GPU is not present, it can execute the operations on the CPU.

3.2.4 Software Requirements

Operating System	Linux-Ubuntu
Programming Language	<ul style="list-style-type: none">• Python - for the algorithms• HTML, CSS and JavaScript - for result visualization
Low-level Libraries	<ul style="list-style-type: none">• CUDA Toolkit• NVIDIA cuDNN• CUDA compute capability > 3.x
Environment	Anaconda - for handling Python packages
Python dependencies	<ul style="list-style-type: none">• TensorFlow• OpenAI-gym• Flask• Numpy• Pillow
IDE/Tool	<ul style="list-style-type: none">• Text Editor• PyCharm (recommended but optional)

3.2.5 Hardware Requirements

Processors	Intel Xeon 2 processors (24 cores)
RAM	32 GB
Storage	500 GB HDD
GPU	NVIDIA FEFORCE GTX
Network	50 Bps or higher

3.2.6 Design Constraints

The System is designed flexibly to experiment with new algorithms and visualize the results. During the program execution, live data can be observed with the help of Web User interface making it convenient for the user, by not forcing to look through log files [27]. Decent bandwidth for the network is required in order to view remote execution results in real-time and interact with the same.

3.2.7 Interfaces

This section describes Software Interface in detail, used for result visualization.

Software Interfaces of the system

- TensorFlow APIs for running mathematical models
- OpenAI gym APIs for collection of environments
- Flask APIs in web server - Domain Classification UI

3.2.6 Non-Functional Requirements

Non-Functional requirements are the requirements [26] that specify criteria used to judge the system operations rather than any specific behaviour which are not concerned with specific functions

- **Reliability:** The system shall always perform correct operation specified by the user
- **Availability:** System and Network connection shall be available at all the time during the training session
- **Security:** The system must be protected from any interruptions, such as killing the running process, unauthorized access to the computer running the code
- **Portability:** System must be able to port to any other platform and operating system. Anaconda package handles all the Python dependencies and sets up portable environment

3.3 Summary

The System Requirement Specifications, constraints of the project have been detailed in this chapter. These include Hardware requirements, software requirements, functional requirements and non-functional requirements.

Chapter 4

High Level Design of Text Classification

Design is the most significant phase in the development of software. In this phase things related to architecture, programming language, data manipulation, data storing with several other aspects are taken into consideration [27]. Here creative knowledge of computer science is converted into application design.

The high-level design is divided into functional and non-functional requirements. The whole system is divided into sub-systems making it easy to design the modules, abstract the data and solve errors internally. The architecture design strategies take into account many design challenges related to performance, convenience of design, faster prototyping and quick modification of source code.

This section describes the challenges in designing the project and justification for many of the design choices made related to the implementation of the project. The issues are critical for developing a proper, expandable and implementable system.

4.1 Design Considerations

This section explains the general constraints that are applicable for the project. These constraints are due to hardware and software limitation and time [28]. These decisions heavily influenced the final product and implementation of the project. This section explains the general constraints and also development methods taken to design the product.

4.1.1 General Constraints

General constraints [28] which need to be considered to use the system are listed below:

- Any user with knowledge of machine learning, Deep learning and TensorFlow can use this system.
- The user should have basic knowledge of deep learning, convolutional network and programming in python

- The user must have detailed knowledge of convolutional network, loss algorithms and machine learning algorithm training.
- The data used for this system is of low resolution due to limited system availability and open source availability.
- The training of the agent is a computationally expensive work, so system is designed to work seamlessly on high end Nvidia GPU
- The agent ones trained can work on low end laptop processor as it requires very less computation during execution.
- The agent configuration remains constant across different environment but the agent need to be trained from the scratch for every environment.

4.1.2 Development methods

The development process involved multiple design process which includes

- The data flow model has been the design method employed for development of the system.
- A data flow model is modelling system based on data transformation that takes place as the data is being processed. The notations used represent functional processing, data stores.
- Data flow models gives the better understanding of how data is associated with the particular process by tracking and providing the documentation.
- Structural diagram is used to design the coding module pattern used to develop modular program structure.
- The system is designed as per system design specification with proper dataflow channels as specified in data flow diagrams.

4.2 Architectural Strategies

This section explains architectural strategies which take a crucial part in design of high level system architecture [28]. These components then translate into a fully functional working system.

4.2.1 Programming Language

Python is the programming language used for prototyping, implementing and testing this project [29]. The reasons for using python as programming language are

- Python is a fast-growing flexible language for implementing development and research work
- Lot of mathematical and computation library (like numpy, Pillow) are easily accessible on python
- Python is based as scripting language and helps in rapid testing of algorithms
- Python supports extensively
- The Tensorflow is an open source framework developed in python and has all its machine learning training interface in python.
- The Flask framework for web based UI.
- Ubuntu the Linux OS is completely integrated with python and python works smoothly on Ubuntu.

4.2.2 User Interface Paradigm

This program provides two web interfaces one to see the training process called Tensorboard that comes with Tensorflow and other is used to see the agent performance after training. Tensorboard is a de-facto training and result analysis tool from Tensorflow. This interface is graph based where training values like loss, network weights and bias along with graph can be observed and analysed [30]. The other interface built for this project is to show the results of training process. The output for the sentence is displayed for analysis. The command-line interface is also provided to change the parameters during the start of the program.

4.2.3 Error Detection and Recovery

The major errors that happen in the program is the change the rand and shape of the data-type when there is change of the environment. The other frequent error that can occur is less GPU ram during training. These types of errors can be controlled by the use of try-catch block and extensive testing [31]. Logging of system action is done at each major step depending on their intensity of error or problem. Even if the error cannot be solved on the

fly, the system is designed to fail gracefully so that other software running on the system are not affected.

4.2.4 Data Storage Management

Data is essential part of any machine learning program. Data generated by the program needs to handle with care and analysed to get maximum of the conversational agent [32]. So, fast in RAM storage and directory structure is constructed for storing. These help in faster access of data for multiple epochs. The directory storage structure makes sure that observation and reward are properly analysed for improving the performance of the system.

4.3 System Architecture

System architecture depicts the complete working of the system in real-time. Each module has its own process work, get its dependent data from other adjacent modules [33]. This software has external module called environment. It gives its observation at every step to the agent. The agent processes the observation using the convolution network. The network returns appropriate action to the agent to forward to the environment. From the action value of the next state, it computes the loss to train the network. The weights of the network are stored in a parameter server so that a common copy is retained across replicated training systems. Overall system architecture is given below in Figure 4.1.

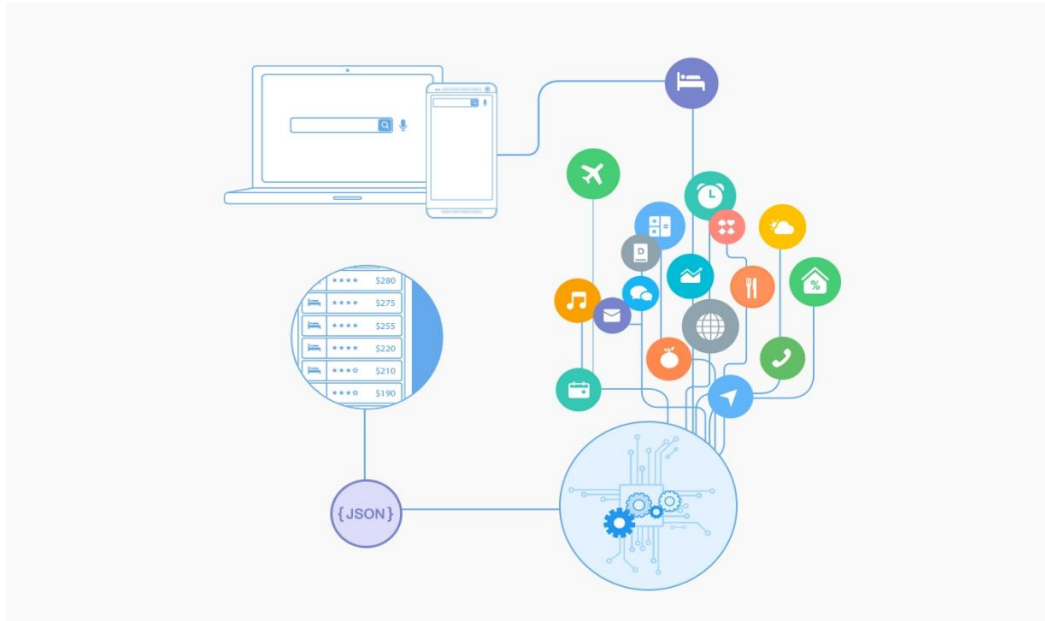


Figure 4.1 System Architecture for Chatbot

There are two web interfaces

1. Tensorboard web interface to analyse the training parameters of the server. This help analyze the weights, loss and other available training parameters during training.
2. The training summary interface allows to analyse the results from agent. These results are in the form of observation, action value, reward and loss.

The system is designed a way that agents can be used for faster training of convolution network by replicated working training method. Here the sentence is fed to the network to classify into different domain [34]. The domains are fixed by the company according the agent where it is being used, then the classified domains can be used to further classify the intend and entity of the agent.

4.4 Data Flow Diagrams

Data Flow Diagram (DFD) graphical representation of the internal working of the design with the flow of data. The data flow is shown to flow from four different types of namely process, data flow, external entity and data store in sequential way. DFD are

composed of many levels for better visual understanding. Each level from Level (0) to Level (n) goes from higher level abstraction of data to deep and clear representation of data flow across the modular design [35]. So, they provide an end user a clear abstract idea regarding the data input and output from each module. DFDs are thus a great way to understand the data flow along with presenting a clear picture of internal working of the whole software.

4.4.1 Level-0 Data Flow Diagram

The Level 0 DFD is a basic overview of system working. There are two modules in Level 0 which are input Sentence and Domain Prediction. Dataset is an internal entity which is responsible for training the network and provide the Domain Prediction process with required observation (current screenshot) and reward (0,1,2) for the previous action. This is given as input to Action Prediction model which executes the network and then recommends most appropriate domain [36]. Figure 4. shows Level 0 DFD the design.

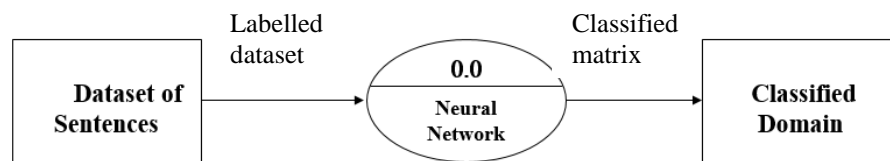


Figure 4.2 Level-0 Data Flow Diagram

4.2.2 Level-1 Data Flow Diagram

Level 1 DFD shows the basic interface of each module in the in level 0. This level has expanded view of Action Prediction from Level 1. Figure 4. is Level-1 DFD for the Level (0.0) [37]. The level has 2 process and a data store to store the observation. Pre-processing module takes the observation and converts it into appropriate format for further processing. The Network Processing process takes the processed observation from the data store and computes the appropriate action considering the reward from the previous action.

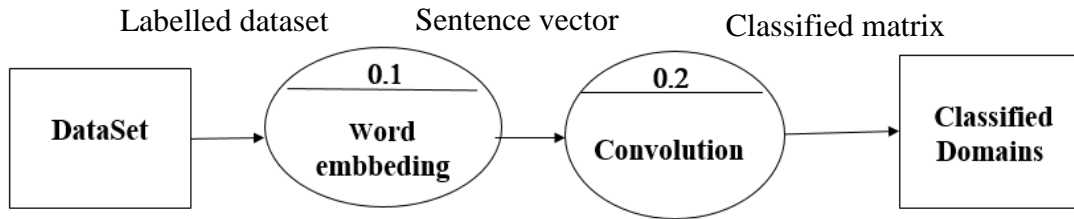


Figure 4.3 Level-1 Data Flow Diagram

4.2.1 Level-2 Data Flow Diagrams

Level 2 DFD contains detailed data flow for each process in Level 1 DFD. Figure 4. shows Level 2 DFD. The input sentence is embedded into a word to vector representation which was already pretrained by the company beforehand (in case any word is not there in the model, it is to be trained again) [38]. Then the convolution of different filter size takes place which in turn generates matrix of probability having domains, which in turn is maxpooled to take maximum possibility and concatenate all the filter size matrix and dropout the compute output at the softmax layer.

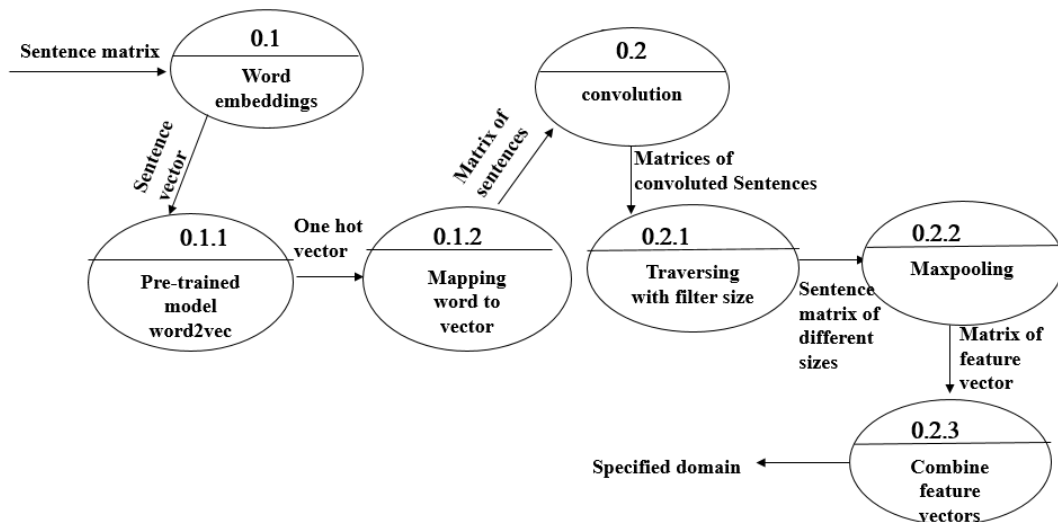


Figure 4.4 Level-2 Data Flow Diagram

The loss computation takes the action value, action and reward for the previous action and then computes appropriate loss from the neural network. This loss is forwarded to weights

training where using neural network training process. The updated weights are then saved in the parameter weights to be used in the next cycle by the neural network.

4.5 Summary

This chapter explained in depth the high-level design with general consideration, architecture strategies, system architecture and in depth process working using Data Flow Diagrams.

Chapter 5

Detailed Design of Text classification

Detailed design explains all the components and subcomponents in the system [39]. Each process in the DFD has its own working principle and had dependent data flow in and results flow out. The detailed design takes into account the necessity of the module along with its purpose in the project. It also gives a clear picture to the designers on how the data processing occurs at each step.

Detailed design also takes into consideration the algorithm used in processing of each module with asymptotic time. This need to be taken into consideration. The following subsection gives the clear insight into each sub module described in the previous chapter.

5.1 Structured Chart of Domain Classification

The structure chart shows the control flow among modules in the system [40]. It explains all the identified modules and the interaction between the modules. It also explains about the identified sub-modules. The structure chart explains the input for each modules and output generated by each module.

In DDQNRL prediction and training are two major sub modules. Prediction does the prediction part of the agent and training trains the DQN network. The prediction gives the observation to action recommender where the action is taken as recommended by the graph or by exploration and exploitation module. The exploration and exploitation module takes action based on exploration probability [41]. The observation along with reward is forwarded to convolution network which comprises of convolution with different filter size and feed-forward layer. The action values for each action is the output. This action values are then used to take the recommended action by action recommender. Structural chart is given in Figure 5.1.

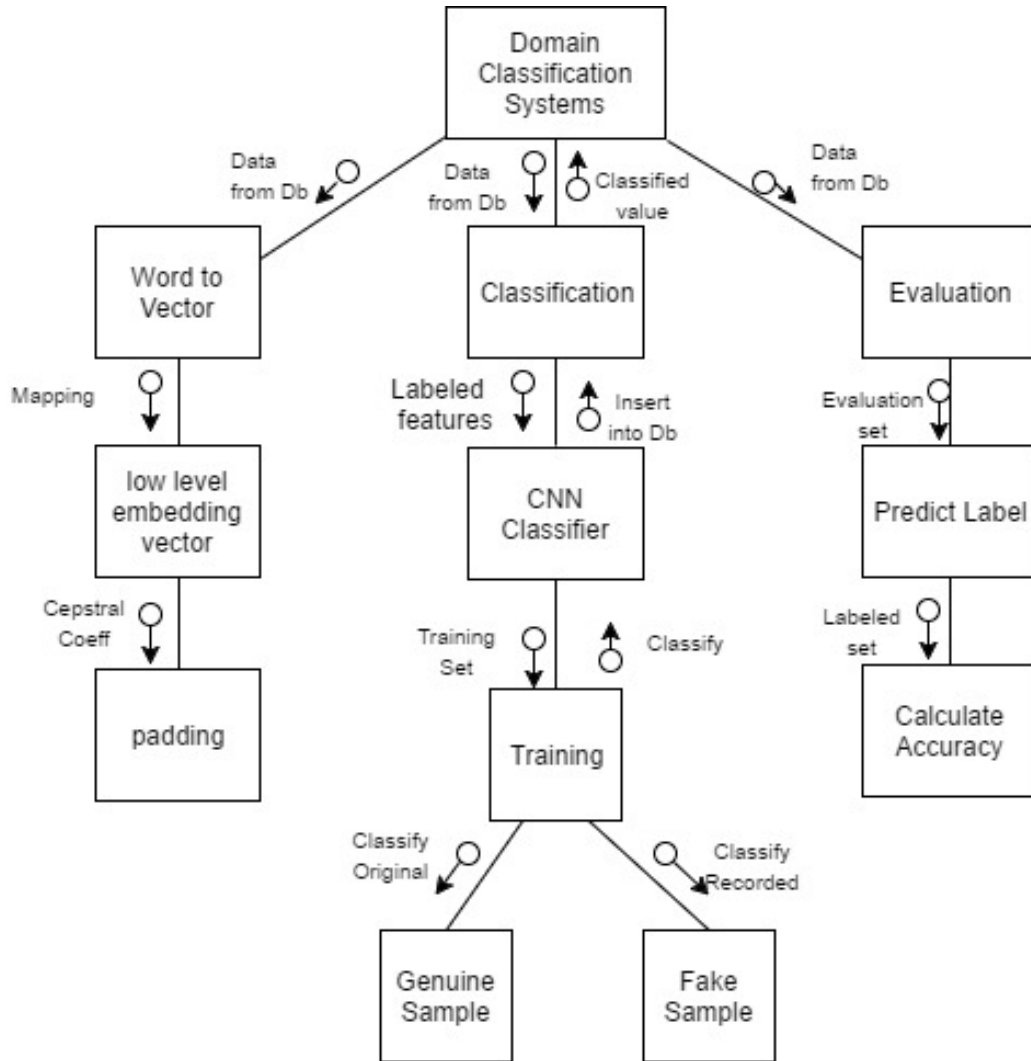


Figure 5.1 Structural Chart of Domain Classification

The other important sub-module is training the graph. This is done in two major stages. First the loss is computed by applying loss. This loss is computed to all the observation by taking the reward into consideration [42]. Then the network is trained on this observation, output, loss set for a fixed number of epochs. This is done by a standard neural network trainer. The updates to the weights are sent to the parameter server for saving. This loop is done for a dataset to train the agent to reach human level performance.

5.2 Functional Description of Modules

This section describes all the internal working of each of the modules in structural diagram including the algorithm for complicated operations.

5.2.1 Learning Loss module

Learning is the extension of Q-Learning algorithm which is extensively popular in reinforcement learning [43]. This algorithm uses the idea of SARSA algorithm where the best value from the next state is used to compute the loss of the network. The network after extensive training can perform greedy action with high accuracy.

- **Purpose:** Classify the sentence into domains which are fixed by the developer.
- **Functionality:** This module takes concatenated input, reward, action value, action and future discount for two consecutive steps and computes loss for the observation action value pair. This loss is very useful as it tells the network where it's taking the wrong action and then correct itself.
- **Output:** domain of the particular network.
- **Flowchart:**

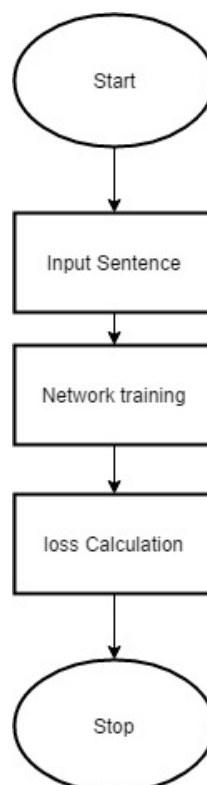


Figure 5.2 Learning Module Flowchart

5.2.2 Convolution Network

This network is used to find the action value for all the action in the environment. This is then used to find the greedy action, which is given to the environment as the action for next step.

- **Purpose:** To compute loss of the action done by the agent after every step.
- **Functionality:** This network is to predict the action value for the next action given the observation and reward for every action. This action value is also used to measure the effectiveness of learning. In the learning phase the values of each state changes rapidly to come to a proper value for each action and value. This process is achieved by using both convolutional then feed-forward network together. The convolution neural network first identifies the useful parts of the image and forms filter values so that values be used for analysis. The feed-forward network generates the notion of all the important part of the image to form perfect image to action value relation.
- **Output:** Action values for each action. The greedy action usually is to take the action with maximum action value
- **Flowchart:**

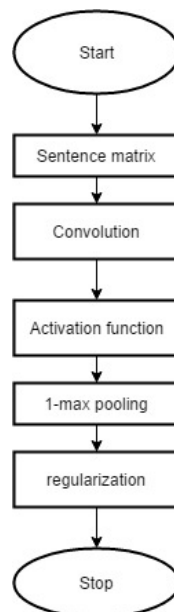


Figure 5.3 Network Flowchart of word embeddings and output

5.3 Summary

This Chapter completely explained each and every aspect towards designing a robust system. Structure diagram and flowchart for each important module make it clear how to proceed with implementation of the project implementation.

Chapter 6

Implementation Details of Domain Classification

Implementation is the most significant part of the project development. Here, the low-level designs are implemented by coding in the specified programming languages. While doing so, it is ensured that the system requirements are met accordingly [45]. Fine details of the design are analysed to ensure the good quality of the system. The selection of platform, programming language, version control, collaboration etc. play important role in this phase.

6.1 Programming Language Selection

For this system, I have chosen Python as the primary programming language [45]. The main reasons being the extensive usage of Python in the areas such as Scientific computing, Machine Learning, Computer Vision and ease of use compared to other programming languages. Other important criteria are as follows:

- Easy to learn and easy to read syntax
- Broad standard library collection
- Extensive support for machine learning algorithms such as scikit-learn, numpy, scipy
- Easy data visualization - matplotlib, plotly
- TensorFlow - A Python-based Open Source library for numerical computation
- Python dependency environment management – Anaconda
- Availability of flexible data structures
- A lightweight Web Framework – Flask

6.2 Platform Selection

The platform selection is necessary because the performance of the system depends on them.

6.2.1 Operating System

Choose Ubuntu (Linux) as our primary operating system to run the programs. The ability to run commands through Terminal and execute bash scripts as per our needs thus making our workflow easier in Ubuntu [46]. Also, it is easier to port the setup in cloud environments, since most of the cloud providers provide Linux virtual machines primarily. Nonetheless, users can try to run the program in Windows since the program is written in Python.

6.2.2 TensorFlow

Choose TensorFlow for numerical computation purposes [47], a high-level library written by Google Inc. The reasons being the following:

- TensorFlow uses graphical model to perform the computations which are easier and efficient in practice.
- It is backed by Google Inc. and a great community of Open Source contributors
- Used by organizations like eBay, DeepMind, Airbnb, ARM, Dropbox etc.
- Multi-GPU support is available for training purposes.
- Collection of Machine Learning and Deep Learning high-level APIs
- Checkpoints of computation model are helpful in training the model for a while, stop to evaluate and resume.
- High performance and out of the box distribution of computations in GPUs
- Ability to visualize the graph itself.
- Visualize scalars, histograms, images, audio, distributions of tensors

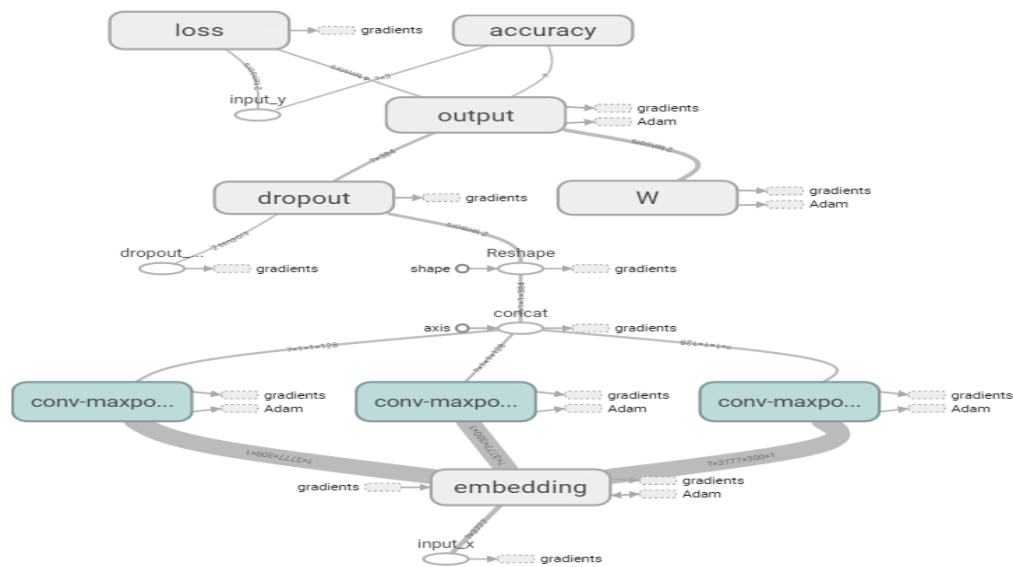


Figure 6.1 TensorBoard graph of Network

The TensorBoard graph of Network is shown in Figure 6.1. It has major 4 components – Convolutional Neural Network, Loss computation, Accuracy and Dropout.

6.2.3 Coding conventions

In this section, coding standards used in this project are discussed [48]. Code conventions are set of guidelines for any set of programming language that is followed within an organization or community. Code conventions make the programs written by many persons look consistent and follow a standard. Also, it makes easier for anyone to read the code without difficulty.

6.2.4 Naming Conventions

For python, followed the PEP-8 conventions, as specified by the standards [49].

General

- Avoid using names which are too general or too wordy. Maintain a good balance between the two.
- While using CamelCase names, capitalize all the letters of an abbreviation.

Packages and Modules

- Package names must be all lowercase.

- When multiple words are needed, separate them with an underscore (eg: concatenated_observation)
- Prefer to stick to single word names

Classes

- Class names should follow uppercase camel convention (eg: GymEnvironment)
- Exception classes should end with phrase “Error”

Global Variables

- Global variables should be all lowercase
- Words in a global variable name should be separated by underscore

Instance Variables

- Instance variable names should be all lowercase
- Words in an instance variable name needs to be separated by underscore
- Non-public instance variables should begin with a single underscore

Methods

- Method names should be all lowercase
- Words in a method name should be separated by underscore
- Non-public method should begin with a single underscore

Method Arguments

- Instance methods should have their first argument named ‘self’.
- Class methods should have their first argument named ‘cls’

Functions

- Function names should be in lowercase
- Words in a function name should be separated by underscore

Constants

- Constant names must be fully capitalized (eg: EXPLORATION_RATE)
- Words in a constant name should be separated by underscore

6.2.5 File Organization

There is no hard and fast rule for organizing file structure. Our codebase is divided into two repositories.

1. DQN algorithm

- This repo holds the code for the DQN algorithm implementation.
- All of the configuration files are kept in root directory
- Code files are kept in 'src/' directory
- Data_helpers.py – cleaning and loading Dataset.
- Text_cnn.py - Class for the tensorflow graph structure
- Train.py – Training the network
- veri.py – verify user statement

2. Domain Classification

- This repo is a web interface for visualizing the results produced by the Network during and after training. It uses Flask web framework
- templates/ - HTML templates for different pages of UI
- runserver.py - Configure and run the server in specified port
- views.py - Classes for different routes

6.2.6 Comments

Comments in Python also follows the PEP 8 convention

- Comments must be complete sentences, if not its first word should be capitalized
- Inline comment is a comment on the same line as the statement.
- Write Docstrings for public modules, functions, classes and methods.

6.4 Difficulties Encountered and Strategies used to tackle

This section mentions the difficulties encountered during the development of the project.

6.4.1 Training Time

Machine Learning algorithms are compute intensive and take a longer time to train a model. This is even more in Deep Learning algorithms due to the presence of many hidden layers. Although, the outcome is more efficient, a few days of training is required for most of the Learning algorithms to learn a relatively simple environment. In order to test the algorithm and tune hyperparameters, it takes a few hours to get the outcome and thereby decide what went wrong.

Deep Learning consists of around ten hyperparameters such as Learning rate, Exploration rate, Replay buffer size etc [47]. Unfortunately, there is no hard and fast rule for these hyperparameter values. It depends on the type of environment and result is unpredictable.

Overcoming this challenge had been difficult. Ran multiple copies of the program parallelly with different hyperparameters by distributing the GPU computation fraction for each of them [48]. At regular intervals, observed the variables such as loss, gradients, weights etc. and decide whether to continue the run or not.

6.5 Summary

This chapter explained about the implementation details of our project, including Platform selection, Programming selection, Code conventions, File structures and challenges faced.

Chapter 7

Software Testing of Domain classification

Software Testing is an essential phase of Software Development process. It is a formalized method for testing the components of programs, or the whole system and thus identify any errors [48]. At this phase, the system is also tested against the requirements, specifications and goals. Test cases are designed to cover a maximum of the possible instances, although One-hundred percent is ideal. It is the responsibility of a programmer either during or after development to write test cases and run them after each change to the application. Test is usually automated to decrease the overhead of engineers. Also, one should give proper valid messages for failures to ease the identification of the errors occurred.

In this chapter, explained the different types of tests have been conducted for different modules of our application. After running each module, integrated and the application is tested as a whole.

7.1 Test Environment

Testing environment for the Network is mainly done using the unittest module for Python. Unit tests are written for the modules, functions and classes. Domain Classification, a visualizing web-interface for our algorithm, is written using Python Flask web framework [48]. It provides a test suite for the different Views in the web application. Also, tests for some helper functions are written using the unittest module.

7.2 Unit Testing

Unit testing, a test engineering technique where individual units of the system are tested separately for their interfaces [48]. It is often automated rather than running manually. This section explains the Unit tests carried out for various units.

7.2.1 Unit tests for Utility module

Utility module contains all the necessary functions required to modify the input given to the network as required.

Table 7.1 Input Processing Test

Sl. No. of Test Case	1
Name of Test Case	Pre-processing input states
Feature being Tested	Pre-processing the input to clean data
Description	This pre-processing cleans the data and saves into processable format.
Sample Input	['Is this the problem?']
Expected Output	Is this the problem?
Actual Output	Is this the problem?
Remarks	Test Successful

The Input Pre-processing module is the initial part. As mentioned before, input to the CNNs are sentences. The required input size for CNNs as per the network architecture is 84x84x4, i.e. last 4 observations. Table 7.1 shows the test conducted for the Input Pre-processing unit.

Table 7.2 Testing Sentence of news Domain

Sl. No. of Test Case	2
Name of Test Case	News domain
Feature being Tested	Sentence of news Domain
Description	Sentence should classify into news domain
Sample Input	Morrisons book second consecutive quarter of sales growth.
Expected Output	1
Actual Output	1
Remarks	Test successful

Table 7.2 shows the input of the sentence which belongs to news domain and should classify into domain giving 2 which gives as news domain.

Network module is the core part of our system. The pre-processed input from the environment is given to this module. Convolution Neural Network, Loss minimization and training happens in this module.

Table 7.3 Test 3 Neural network construction

Sl. No. of Test Case	3
Name of Test Case	Neural network building
Feature being Tested	Construction with size of neural network
Description	Neural network is converted into respective matrix multiplication while propagating through each layer.
Sample Input	The size of the input
Expected Output	The network in TensorFlow object with number of actions as output.
Actual Output	TensorFlow object with 6 output.
Remarks	Successful construction of the network.

Table 7.3 shows the unit test for the Neural network building part. Architecture for the Convolution Neural Network is given in Fig 5.3. Thus, output layer of the CNN has to be a full connected layer with 3 outputs. Each output corresponds to the Value for each action.

Table 7.4 Testing Sentence of Control domain

Sl. No. of Test Case	4
Name of Test Case	Control domain
Feature being Tested	Sentence of Control domain.
Description	Sentence should classify into Control domain
Sample Input	Increase the temperature.
Expected Output	0-control domain
Actual Output	0-control domain
Remarks	Successful retrieval of current state

Table 7.4 shows the input of the sentence which belongs to Control domain and should classify into domain giving 2 which gives as Control domain.

Table 7.5 Unit test for Training process of the network

Sl. No. of Test Case	5
Name of Test Case	Training network
Feature being Tested	Training process of the network
Description	Training process is the most important process of the network as in this phase the network learns from its mistakes and reward that it obtains.
Sample Input	Error computed from Network
Expected Output	Delta values for each weight
Actual Output	Delta values for each weight
Remarks	Correct update of hidden layer weights during training

Table 7.5 shows the unit test for the training process of the Network. Output is observed continuously so that delta values for weights values do not explode or vanish.

Table 7.6 Testing Sentence of news Domain

Sl. No. of Test Case	6
Name of Test Case	Sentence of different domain
Feature being Tested	Domain which is not been defined
Description	Sentence which is not related to any domain is being tested.
Sample Input	Feeling good.
Expected Output	0
Actual Output	0
Remarks	Test Unsuccessful

Table 7.6 shows the input of the sentence which belongs to different domain, not defined by the user gets classified into control.

Table 7.7 Testing Sentence of weather domain

Sl. No. of Test Case	7
Name of Test Case	weather domain
Feature being Tested	Sentence of weather domain
Description	Sentence should classify into Weather domain
Sample Input	Is it drizzling today?
Expected Output	0- weather
Actual Output	0-weather
Remarks	Test Successful

Table 7.7 shows the input of the sentence which belongs to Weather domain and should classify into domain giving 2 which gives as Weather domain.

Table 7.8 Testing Dataset with other ascii input

Sl No. of Test Case:	8
Name of Test:	Different language data
Feature being Tested:	Dataset with other ascii input
Sample Input:	Chinese input
Expected Output:	unexpected end of input; is count incorrect or file otherwise damaged?
Actual Output:	unexpected end of input; is count incorrect or file otherwise damaged?
Remarks:	Test Successful

Table 7.8 shows the input of the sentence which does not have ascii value, since the network is not trained on it, neither it is there in the vocabulary it will give an error.

Table 7.9 Testing Different domain sentence

Sl No. of Test Case:	9
Name of Test:	Different domain sentence
Feature being Tested:	Sentence with different domain not defined
Sample Input:	Where was Akbar born?
Expected Output:	Error
Actual Output:	0-control domain
Remarks:	Test Unsuccessful

Table 7.9 shows the input of the sentence which belongs to different domain not defined by the user and should classify into domain giving 2 which gives as Weather domain.

Table 7.10 Testing for Words not in vocab

Sl No. of Test Case:	10
Name of Test:	Vocab problem
Feature being Tested:	Words not in vocab
Sample Input:	pneumonoultramicroscopicsilicovolcanoconiosis
Expected Output:	invalid vector on line
Actual Output:	invalid vector on line
Remarks:	Test Successful

Table 7.10 shows the input of the words in the Sentence which cannot be represented as vector hence gives error as invalid on vector line.

Table 7.11 Testing Sentence of news Domain

Sl. No. of Test Case	11
Name of Test Case	News domain
Feature being Tested	Sentence of news Domain

Description	Sentence should classify into news domain
Sample Input	Open the news channel which shows current news?
Expected Output	0
Actual Output	1
Remarks	Test Unsuccessful

Table 7.11 has the input of sentence which is ambiguous to the network in which domain it should classify, news or control domain.

7.3 Integration Testing

Integration Testing is a software testing phase where the individual modules of the system are combined and tested as a group.

7.3.1 Integration testing for Domain classification

Table 7.12 Integration testing for Domain Classification

Sl. No. of Test Case	12
Name of Test Case	Reading files from storage and displaying it on Domain Classification
Feature being Tested	File retriever and HTTP server
Description	Data written by file writer from game execution is read by file retriever for displaying it on web interface. Retrieved data is passed to web server which converts data to HTML format for web interface.
Sample Input	Execution data from file system
Expected Output	HTML format data for web interface
Actual Output	HTML format data for web interface
Remarks	Successfully accessing web interface via web browser

Table 7.12 shows the integration testing for reading files from storage and displaying it on Domain Classification. The written files are tested on Domain Classification and ensured that they are converted to HTML format.

7.4 System Testing

System Testing is a stage in software testing where a complete and integrated system is tested. The main purpose of this stage is to evaluate the system against specified requirements.

Table 7.13 System Testing

Sl. No. of Test Case	13
Name of Test Case	Breakout environment execution
Feature being Tested	Whole working of system from input, file writing, Domain Classification and TensorBoard
Description	Whole system is tested end-to-end and made sure that if system works perfectly with no errors and exception. This is useful for testing the training time of the system and knowing the real-time failures during execution.
Sample Input	Standard input in configuration file
Expected Output	Agent learning the best action for a specific environment
Actual Output	Agent learns to play at super human performance
Remarks	Better performance can be achieved with extensive training and hyper parameter tuning

Table 7.13 shows the system testing. This took “Breakout-v0” environment for testing the whole system. After a few days of training, the agent has learnt to play the game at super-human performance.

7.5 Summary

This chapter explained about the various tests carried out for this project, including Test Environment, Unit Testing, Integration testing and System testing.

Chapter 8

Experimental Results and Analysis for Classification Network

Experimental result analysis is an important part to measure the success of the project. Here results are observed with a carefully and changes in execution is made to get the best result. The analysis part for this project includes making sure that evaluation metrics are optional for a given game and training process is done in short time exploiting maximum available system resource. This project has several evaluation metrics and other parameters that are tuned during its execution. The agent is trained in multiple environment to test its ability to work with multiple environment. The evaluation metrics of all these environments give us comprehensive insight into the agent training and testing. This chapter includes the evaluation metrics and performance of the agent for each of the trained environment.

8.1 Performance analysis of results

The preparation script composes outlines to a yield catalog, and by guiding TensorBoard toward that registry can envision the diagram and the synopses made.

The dataset consists of 1GB of data with about more than 200 sentences in each domain i.e., Weather, News and Control. Running the Training with default parameters (128-dimensional embeddings, channel sizes of 3, 4 and 5, dropout of 0.5, Dropout is a technique where randomly selected neurons are ignored during training, and 128 channels for every channel measure) brings about the accompanying misfortune and exactness plots (blue is preparing information, red is 10% dev information) shown in Figure 8.1.

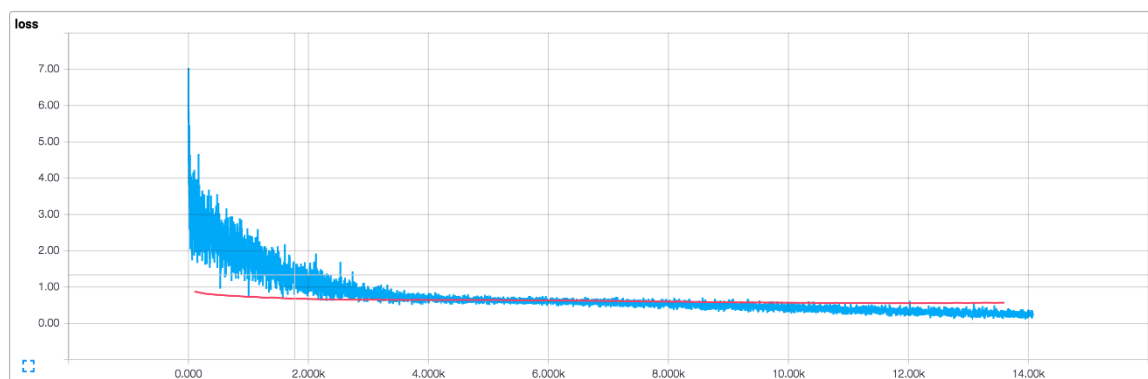


Figure 8.1 Graph of loss vs iterations.

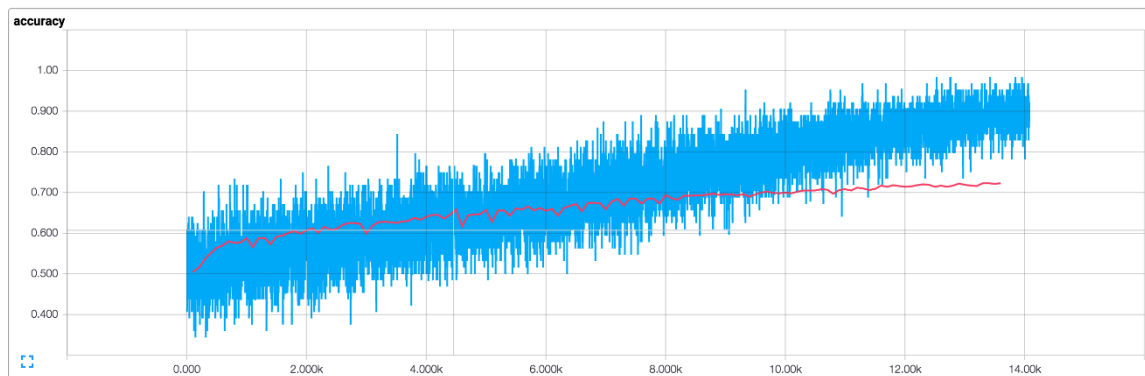


Figure 8.2 Graph of Accuracy vs iterations.

There are a few things that emerge:

- The preparation measurements are not smooth since it utilized little group sizes. On the off chance that utilized bigger clusters (or assessed all in all preparation set) would get a smoother blue line.
- Since dev accuracy is fundamentally beneath preparing exactness it appears like our system is overfitting the preparation information, proposing that require more information (the MR dataset is little), more grounded regularization, or less model parameters. For instance, different explored avenues regarding including extra L2 punishments for the weights at the last layer and could knock up the accuracy to 76%.
- The preparation misfortune and exactness begins altogether beneath the dev measurements because of dropout connected to it, shown in figure 8.2.

Evaluation metrics

Evaluation metrics are used to measure the performance of the application detection system. With the help of these metrics, the parameters are analysed to get the best result for the system.

This project uses the following evaluation metrics to improve performance of its agent.

- **Accuracy:** It is used to check the softmax layer the percentage of correct predictions.
- **Loss:** It is a summation of the errors made for each example in training or validation sets.

Application	SVM	CNN
Accuracy	87%	96%
Loss	0.057	0.021

8.2 Summary

The results obtained consist of all the results expected at the beginning of the project. The model exhibits accuracy of 96% with the spoken language systems with some loss 0.021.

Chapter 9

Conclusion and Future Enhancement

Utilized the repetitive neural system for logical area arrangement in a multi-turn multi-area SLU session. The proposed CNN approach specifically utilizes the past model expectation as extra elements for the present turn. The subsequent show outflanks with relevant elements by huge edges.

With the results and analysis following conclusions are made:

- The agent successfully learned the state space for all the given environment.
- The performance of the agent completely depended on the training process and the network hyper-parameters.
- The tuning of hyper-parameters requires multiple failed training process, which is computation and time expensive.
- The fixed network can be trained on different environment and able to learn each environment successfully
- The network is programmed to maximize the utilization of GPU on every node.
- TensorFlow (Machine Learning framework used for implementation) is capable of abstracting the GPU and CPU and form a uniform computation platform for seamless implementation.
- The use of GPUs accelerated the implementation and working by ten-fold
- The project implementation has multiple distributed agents and thus can be trained simultaneously. This is very useful as the agent converges to the best action in very short time.
- The distributed agent if deployed on the same system can maximize system utilization and exploit the best of what is available.

9.1 Limitations of the Project

- The network is unstable during training if the hyper-parameters such as Learning rate or Initialization of weights are not properly set.

- The user has to analyze the log to make sure there is a stable learning process and loss isn't exploding or vanishing.
- Exploration ratio needs to be set by the user by analyzing the environment state space
- Requires a minimum knowledge of Deep Learning and Reinforcement Learning to work with the framework developed
- The distributed agent training doesn't have a centralized logging system.
- Failure in distributed agent training is hard to detect.

9.2 Future Enhancements

Here are a few valuable activities that may enhance the execution of the model:

- Instate the embeddings with pre-prepared word2vec vectors. To make this work have to utilize 300-dimensional embeddings and instate them with the pre-prepared qualities.
- Oblige the L2 standard of the weight vectors in the last layer, much the same as the first paper. You can do this by characterizing another operation that updates the weight values after each preparation step.
- Add L2 regularization to the system to battle overfitting, likewise try different things with expanding the dropout rate.
- Include histogram synopses for weight updates and layer activities and picture them in TensorBoard.

9.3 Learning from project

The project helped to understand several types of machine learning models and how to develop a neural network on TensorBoard, which is platform for fast processing of neural networks. Understood different models created by the company and worked on the web UI of these models to take input and test the models.

References

- [1] H. Chae, C. Mook Kang, B. Kim, J. Kim, Chung Choo Chung and Jun Won Choi, "Autonomous Braking System via Deep Reinforcement Learning," CoRR, vol. 1702.02302v1 [cs.AI], 2017, pp. 44-67.
- [2] Nigam, Kamal, et al. "Text classification from labeled and unlabeled documents using EM." Machine learning 39.2 ,2000, pp. 103-134.
- [3] Y. Bengio, "Learning Deep Architectures for AI, Foundations and Trends in Machine Learning", vol. 2, no. 1, 2009, pp. 120-127.
- [4] G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang and W. Zaremba, "OpenAI Gym," OpenAI, Available: <https://gym.openai.com/>, 5 June 2016.
- [5] H. Lee, R. Grosse, R. Ranganath and A. Y. Ng, "Unsupervised Learning of Hierarchical Representations with Convolutional Deep Belief Networks," Proceedings of the 26th Annual International Conference on Machine Learning, 2009, pp. 609-616.
- [6] Dean, Jeffery, "TensorFlow: A System for Large-Scale Machine Learning," Google Brain, Available: <https://www.tensorflow.org>. 2015.
- [7] Theano Development Team, "Theano: A Python framework for fast computation of mathematical expressions," [Online]. Available: <http://deeplearning.net/software/theano/>, 2008.
- [8] M. Khamassi and C. Tzafestas, "Active exploration in parameterized reinforcement learning," CoRR, vol. abs/1610.01986, 2016, pp. 112-122.
- [9] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra and M. A. Riedmiller, "Playing Atari with Deep Reinforcement Learning," CoRR, vol. abs/1312.5602, 2013, pp. 45-81.

- [10] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. P. Lillicrap, T. Harley, D. Silver and K. Kavukcuoglu, “*Asynchronous Methods for Deep Reinforcement Learning*,” *CoRR*, vol. abs/1602.01783, 2016, pp.87-101.
- [11] Danijar Hafner, “*Deep Reinforcement Learning From Raw Pixels in Doom*,” *CoRR*, vol. abs/1610.02164, 2016, pp.2-3.
- [12] H.-T. Cheng, L. Koc, J. Harmsen, T. Shaked, T. Chandra, H. Aradhye, G. Anderson, W. Chai, M. Ispir, R. Anil, Z. Haque, L. Hong, V. Jain, X. Liu and H. Shah, “*Wide & Deep Learning for Recommender Systems*,” *CoRR*, vol. abs/1606.07792, 2016, pp.4-5.
- [13] N. Jaques, Shixiang Gu, R. E. Turner and D. Eck, “*Tuning Recurrent Neural Networks with Reinforcement Learning*,” *CoRR*, vol. abs/1611.02796, 2016, pp.6-10.
- [14] D. Silver, H. van Hasselt and M. Hessel, “*The Predictron: End-To-End Learning and Planning*,” *CoRR*, vol. abs/1612.08810, 2016, pp.2-4.
- [15] A. Barreto, R. Munos, T. Schaul and D. Silver, “*Successor Features for Transfer in Reinforcement Learning*,” *CoRR*, vol. abs/1606.05312, 2016, pp.1-29.
- [16] I. Sutskever, O. Vinyals and Q. V. Le, “*Sequence to Sequence Learning with Neural Networks*,” *CoRR*, vol. abs/1409.3215, 2014, pp.4-2.
- [17] B. Bakker, “*Reinforcement Learning with Long Short-Term Memory*,” in *In NIPS*, Cambridge, Massachusetts: MIT Press, 2002, pp.1475-1482.
- [18] K. W. Mathewson and P. M. Pilarski, “*Simultaneous Control and Human Feedback in the Training of a Robotic Agent with Actor-Critic Reinforcement Learning*,” *CoRR*, vol. abs/1606.06979, 2016, pp.22-90.
- [19] J. Veness, K. Siong Ng, M. Hutter and D. Silver, “*Reinforcement Learning via AIXI Approximation*,” *CoRR*, vol. abs/1007.2049, 2010, pp.45-81.

- [20] T. Shankar, S. K. Dwivedy and P. Guha, “*Reinforcement Learning via Recurrent Convolutional Neural Networks*,” *CoRR*, vol. abs/1701.02392, 2017, pp.34-61.
- [21] Y. Duan, J. Schulman, X. Chen, P. L. Bartlett, I. Sutskever and P. Abbeel, “*RL2: Fast Reinforcement Learning via Slow Reinforcement Learning*,” *CoRR*, vol. abs/1611.02779, 2016, pp.1-30.
- [22] C. Fernando, D. Banarse, C. Blundell, Y. Zwols, D. Ha, A. A. Rusu, A. Pritzel and D. Wierstra, “*PathNet: Evolution Channels Gradient Descent in Super Neural Networks*,” *CoRR*, vol. abs/1701.08734, 2017, pp.23-60.
- [23] I. Bello, H. Pham, Q. V. Le, M. Norouzi and S. Bengio, “*Neural Combinatorial Optimization with Reinforcement Learning*,” *CoRR*, vol. abs/1611.09940, 2016, pp.35-60.
- [24] A. Nair, P. Srinivasan and D. Silver, “*Massively Parallel Methods for Deep Reinforcement Learning*,” *CoRR*, vol. abs/1507.04296, 2015, pp.56-94.
- [25] S. Sharifzadeh, I. Chiotellis, R. Triebel and D. Cremers, “*Learning to Drive using Inverse Reinforcement Learning and Deep Q-Networks*,” *NIPS workshop on Deep Learning for Action and Interaction*, vol. abs/1612.03653, 2016, pp.2-8.
- [26] M. Denil, P. Agrawal, T. D Kulkarni, T. Erez, P. Battaglia and Nando de Freitas, “*Learning to Perform Physics Experiments via Deep Reinforcement Learning*,” *CoRR*, vol. abs/1611.01843, 2016, pp.4-10.
- [27] Hado van Hasselt, A. Guez and D. Silver, “*Deep Reinforcement Learning with Double Q-learning*,” *AAAI 2016*, vol. abs/1509.06461, 2016, pp.100-121.
- [28] D. Yogatama, P. Blunsom, C. Dyer, E. Grefenstette and W. Ling, “*Learning to Compose Words into Sentences with Reinforcement Learning*,” *CoRR*, vol. abs/1611.09100, 2016, pp.12-44.

- [29] S. Praveen Bangaru, J. Suhas and B. Ravindran, “*Exploration for Multi-task Reinforcement Learning with Deep Generative Models*,” *NIPS Deep Reinforcement Learning Workshop 2016, Barcelona*, vol. abs/1611.09894, 2016, pp.3-8.
- [30] B. Dhingra, L. Li and X. Li, “*End-to-End Reinforcement Learning of Dialogue Agents for Information Access*,” *CoRR*, vol. abs/1609.00777, 2016, pp.211-300.
- [31] O. Anschel, N. Baram and N. Shimkin, “*Averaged-DQN: Variance Reduction and Stabilization for Deep Reinforcement Learning*,” *CoRR*, vol. abs/1611.01929, 2016, pp.34-45.
- [32] J. Heinrich and D. Silver, “*Deep Reinforcement Learning from Self-Play in Imperfect-Information Games*,” *CoRR*, vol. abs/1603.01121, 2016, pp.12-63
- [33] G. Hinton, N. Srivastava and K. Swersky, “*Lecture 6a overview of mini-batch gradient descent*,” Available: http://www.cs.toronto.edu/~tijmen/csc321/slides/lecture_slides_lec6.pdf. 2012, pp.33-43.
- [34] X. Glorot and Y. Bengio, “*Understanding the difficulty of training deep feedforward neural networks*,” *Proc. AISTATS*, vol. 9, 2010, pp.249–256.
- [35] R. Bellman, “*A Markovian Decision Process*,” *Indiana University Mathematics Journal*, vol. 6, no. 4, 1957, pp. 679-684.
- [36] D. P. Bertsekas, *Dynamic Programming and Optimal Control*, 3rd Edition ed., vol. II, The M.I.T. Press, 2011, pp. 332-337.
- [37] R. Sutton, “*Learning to predict by the methods of temporal differences*,” *Machine Learning*, Addison-Wesley Publishing Company, vol. 3, 2000, pp. 9-44.
- [38] M. Niranjan and G. A. Rummery, “*Online Q-Learning using Connectionist Systems*,” Available: <https://goo.gl/dHpjNz>.
- [39] I. Sommerville, *Software Engineering*, 9th Edition ed., vol. ISBN: 9780137035151, Addison-Wesley Publishing Company, 2010, pp.123-150

- [40] SciPy developers, “*Scientific Computing Tools for Python*,” Community library project, 2001. Available: <https://www.scipy.org>. 2010
- [41] Continuum analytics, “*Anaconda (Python Distribution)*,” 2013. Available: <https://www.continuum.io/>. [2011].
- [42] A. Ronacher, “*Flask - A Python Microframework*,” 1 April 2010. Available: <http://flask.pocoo.org/>. [2011].
- [43] Google Research, “*TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems*,” 9 November 2015. Available: <https://goo.gl/2vHVUz>. [2011].
- [44] M. G. Bellemare, Y. Naddaf, J. Veness and M. Bowling, “*The Arcade Learning Environment: An Evaluation Platform for General Agents*,” *Journal of Artificial Intelligence Research*, vol. 47, 2014, pp. 253-279.
- [45] R. L. Glass, *Facts and Fallacies of Software Engineering*, Addison Wesley, 2003, pp. 143-161.
- [46] Python Software Foundation (PSF), “*PEP 8 -- Style Guide for Python Code*,” Available: <https://www.python.org/dev/peps/pep-0008/>. [2017].
- [47] R. Patton, *Software Testing*, 2nd Edition ed., Pearson Education, 2006, pp. 103-144
- [48] Python Software Foundation (PSF), “*unittest - Unit Testing Framework - Python*,” Available: <https://docs.python.org/3/library/unittest.html>.
- [49] JetBrains, “*PyCharm*,” July 2010. Available: <https://www.jetbrains.com/pycharm/>. [2017].
- [50] M. Niranjan and G. A. Rummery, “*Online Q-Learning using Connectionist Systems*,” 1994. Available: <https://goo.gl/dHpjNz>. [2010].

Appendices

Appendix A - Screenshots of Training and data

Figure A.1 shows the dataset from the different domains. It has the sentence in the text format.

```

saumya@saumya-DT: ~/cldom/domain-classification/data/SentenceCorpus/weather
saumya@saumya-DT: ~/cldom/domain-classification/data/SentenceCorpus/control$ ls
control_100.txt control_115.txt control_12.txt control_17.txt control_31.txt control_46.txt control_60.txt control_75.txt control_8.txt
control_101.txt control_116.txt control_130.txt control_18.txt control_32.txt control_47.txt control_61.txt control_76.txt control_90.txt
control_102.txt control_117.txt control_131.txt control_19.txt control_33.txt control_48.txt control_62.txt control_77.txt control_91.txt
control_103.txt control_118.txt control_132.txt control_1.txt control_34.txt control_49.txt control_63.txt control_78.txt control_92.txt
control_104.txt control_119.txt control_133.txt control_20.txt control_35.txt control_4.txt control_64.txt control_79.txt control_93.txt
control_105.txt control_120.txt control_134.txt control_21.txt control_36.txt control_50.txt control_65.txt control_80.txt control_94.txt
control_106.txt control_121.txt control_135.txt control_22.txt control_37.txt control_51.txt control_66.txt control_81.txt control_95.txt
control_107.txt control_122.txt control_136.txt control_23.txt control_38.txt control_52.txt control_67.txt control_82.txt control_96.txt
control_108.txt control_123.txt control_137.txt control_24.txt control_39.txt control_53.txt control_68.txt control_83.txt control_97.txt
control_109.txt control_124.txt control_138.txt control_25.txt control_40.txt control_54.txt control_69.txt control_84.txt control_98.txt
control_110.txt control_125.txt control_139.txt control_26.txt control_41.txt control_55.txt control_70.txt control_85.txt control_99.txt
control_111.txt control_126.txt control_140.txt control_27.txt control_42.txt control_56.txt control_71.txt control_86.txt control_9.txt
control_112.txt control_127.txt control_141.txt control_28.txt control_43.txt control_57.txt control_72.txt control_87.txt
control_113.txt control_128.txt control_142.txt control_29.txt control_44.txt control_58.txt control_73.txt control_88.txt
control_114.txt control_129.txt control_143.txt control_30.txt control_45.txt control_59.txt control_74.txt control_89.txt
saumya@saumya-DT: ~/cldom/domain-classification/data/SentenceCorpus/control$ cd ..
saumya@saumya-DT: ~/cldom/domain-classification/data/SentenceCorpus/news$ ls
news_100.txt news_18.txt news_26.txt news_34.txt news_42.txt news_50.txt news_59.txt news_67.txt news_75.txt news_83.txt news_91.txt news_9.txt
news_10.txt news_19.txt news_27.txt news_35.txt news_43.txt news_51.txt news_5.txt news_68.txt news_76.txt news_84.txt news_92.txt
news_11.txt news_1.txt news_28.txt news_36.txt news_44.txt news_52.txt news_60.txt news_69.txt news_77.txt news_85.txt news_93.txt
news_12.txt news_20.txt news_29.txt news_37.txt news_45.txt news_53.txt news_61.txt news_70.txt news_78.txt news_86.txt news_94.txt
news_13.txt news_21.txt news_30.txt news_38.txt news_46.txt news_54.txt news_62.txt news_71.txt news_79.txt news_87.txt news_95.txt
news_14.txt news_22.txt news_31.txt news_39.txt news_47.txt news_55.txt news_63.txt news_72.txt news_80.txt news_88.txt news_96.txt
news_15.txt news_23.txt news_32.txt news_40.txt news_48.txt news_56.txt news_64.txt news_73.txt news_81.txt news_89.txt news_97.txt
news_16.txt news_24.txt news_33.txt news_41.txt news_49.txt news_57.txt news_65.txt news_74.txt news_82.txt news_90.txt news_98.txt
news_17.txt news_25.txt news_34.txt news_42.txt news_50.txt news_58.txt news_66.txt news_75.txt news_83.txt news_91.txt news_99.txt
saumya@saumya-DT: ~/cldom/domain-classification/data/SentenceCorpus/news$ cd ..
saumya@saumya-DT: ~/cldom/domain-classification/data/SentenceCorpus/weather$ ls
weather_10.txt weather_19.txt weather_27.txt weather_35.txt weather_43.txt weather_51.txt weather_5.txt weather_68.txt weather_76.txt weather_84.txt
weather_11.txt weather_1.txt weather_28.txt weather_36.txt weather_44.txt weather_52.txt weather_60.txt weather_69.txt weather_77.txt
weather_12.txt weather_20.txt weather_29.txt weather_37.txt weather_45.txt weather_53.txt weather_61.txt weather_70.txt weather_78.txt
weather_13.txt weather_21.txt weather_2.txt weather_38.txt weather_46.txt weather_54.txt weather_62.txt weather_71.txt weather_79.txt
weather_14.txt weather_22.txt weather_30.txt weather_39.txt weather_47.txt weather_55.txt weather_63.txt weather_72.txt weather_80.txt
weather_15.txt weather_23.txt weather_31.txt weather_40.txt weather_48.txt weather_56.txt weather_64.txt weather_73.txt weather_81.txt
weather_16.txt weather_24.txt weather_32.txt weather_41.txt weather_49.txt weather_57.txt weather_65.txt weather_74.txt weather_82.txt
weather_17.txt weather_25.txt weather_33.txt weather_42.txt weather_50.txt weather_58.txt weather_66.txt weather_75.txt weather_83.txt
weather_18.txt weather_26.txt weather_34.txt weather_43.txt weather_51.txt weather_59.txt weather_67.txt weather_76.txt weather_84.txt
saumya@saumya-DT: ~/cldom/domain-classification/data/SentenceCorpus/weather$

```

Figure A.1 Dataset of three domains

Figure A.2 shows the training Process of Domain Classification Neural Network.it prints the loss, accuracy and Step.

```

saumya@saumya-DT: ~/cdom/domain-classification
2017-04-24T11:49:32.172900: step 59, loss 0.0454456, acc 1
2017-04-24T11:49:32.186908: step 60, loss 0.0258382, acc 1
2017-04-24T11:49:32.201055: step 61, loss 0.100184, acc 0.984375
2017-04-24T11:49:32.215389: step 62, loss 0.0115004, acc 1
2017-04-24T11:49:32.228904: step 63, loss 0.121495, acc 0.96875
2017-04-24T11:49:32.243093: step 64, loss 0.146781, acc 0.953125
2017-04-24T11:49:32.257425: step 65, loss 0.0846625, acc 0.961538
2017-04-24T11:49:32.271523: step 66, loss 0.0261515, acc 1
2017-04-24T11:49:32.285341: step 67, loss 0.0919226, acc 0.96875
2017-04-24T11:49:32.299939: step 68, loss 0.120098, acc 0.96875
2017-04-24T11:49:32.313635: step 69, loss 0.101432, acc 0.953125
2017-04-24T11:49:32.328060: step 70, loss 0.135941, acc 0.884615
2017-04-24T11:49:32.342720: step 71, loss 0.0687295, acc 0.96875
2017-04-24T11:49:32.357014: step 72, loss 0.0749387, acc 0.984375
2017-04-24T11:49:32.370545: step 73, loss 0.0231191, acc 1
2017-04-24T11:49:32.385069: step 74, loss 0.140482, acc 0.953125
2017-04-24T11:49:32.398890: step 75, loss 0.0134401, acc 1
2017-04-24T11:49:32.413421: step 76, loss 0.0871272, acc 0.96875
2017-04-24T11:49:32.427332: step 77, loss 0.0298699, acc 1
2017-04-24T11:49:32.441775: step 78, loss 0.0604385, acc 0.953125
2017-04-24T11:49:32.455493: step 79, loss 0.0715668, acc 0.96875
2017-04-24T11:49:32.471112: step 80, loss 0.126138, acc 0.961538
2017-04-24T11:49:32.486720: step 81, loss 0.0574414, acc 0.984375
2017-04-24T11:49:32.503843: step 82, loss 0.08662, acc 0.96875
2017-04-24T11:49:32.517368: step 83, loss 0.0862922, acc 0.953125
2017-04-24T11:49:32.531029: step 84, loss 0.0389972, acc 0.984375
2017-04-24T11:49:32.544186: step 85, loss 0.160735, acc 0.923077
2017-04-24T11:49:32.558829: step 86, loss 0.0169992, acc 1
2017-04-24T11:49:32.573341: step 87, loss 0.0137734, acc 1
2017-04-24T11:49:32.588173: step 88, loss 0.112546, acc 0.953125
2017-04-24T11:49:32.602187: step 89, loss 0.107755, acc 0.96875
2017-04-24T11:49:32.616593: step 90, loss 0.0315932, acc 1
2017-04-24T11:49:32.631420: step 91, loss 0.0500717, acc 0.96875
2017-04-24T11:49:32.646212: step 92, loss 0.0190676, acc 1
2017-04-24T11:49:32.660085: step 93, loss 0.0382653, acc 0.96875
2017-04-24T11:49:32.674699: step 94, loss 0.0814954, acc 0.984375
2017-04-24T11:49:32.689263: step 95, loss 0.128991, acc 0.961538
2017-04-24T11:49:32.704901: step 96, loss 0.01724, acc 1
2017-04-24T11:49:32.718941: step 97, loss 0.129081, acc 0.96875
2017-04-24T11:49:32.732624: step 98, loss 0.0145862, acc 1
2017-04-24T11:49:32.746745: step 99, loss 0.0298404, acc 1
2017-04-24T11:49:32.761783: step 100, loss 0.0361861, acc 1

Evaluation:

```

Figure A.2 Training Process

Figure A.3 shows the training Process of Domain Classification Neural Network. This is the evaluation of the data in the dataset and output in the last matrix.

```

saumya@saumya-DT: ~/cdom/domain-classification
2017-04-24T11:49:32.704801: step 96, loss 0.01724, acc 1
2017-04-24T11:49:32.718941: step 97, loss 0.129081, acc 0.96875
2017-04-24T11:49:32.732624: step 98, loss 0.0145862, acc 1
2017-04-24T11:49:32.746745: step 99, loss 0.0298404, acc 1
2017-04-24T11:49:32.761783: step 100, loss 0.0361861, acc 1

Evaluation:
2017-04-24T11:49:32.798549: step 100, loss 0.0772153, acc 0.967742
, [[ 93  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0]
 [791 362  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0]
 [122  5  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0]
 [491 362  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0]
 [323 283 324 75 227 325 326 327 328 128 329 330  0  0  0  0  0  0  0]
 [193 150 165  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0]
 [ 20 155 49 27 53  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0]
 [492 23 493 343 484 128 495 496  0  0  0  0  0  0  0  0  0  0  0]
 [737 738 40 679 739 488  0  0  0  0  0  0  0  0  0  0  0  0  0  0]
 [ 75 45 76 77 43 45 78 79 80  0  0  0  0  0  0  0  0  0  0  0]
 [280 281 13 18 120 151  0  0  0  0  0  0  0  0  0  0  0  0  0  0]
 [256 85 211  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0]
 [589 18 362 40 122  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0]
 [256 206 211  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0]
 [658  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0]
 [158 159 160 161 162 163  0  0  0  0  0  0  0  0  0  0  0  0  0  0]
 [214  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0]
 [ 49 27 426  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0]
 [216  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0]
 [ 43 27 44 639  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0]
 [ 84 45 87 88 89  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0]
 [183 184  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0]
 [197 753 754 98 755 756 13 757 758 759  0  0  0  0  0  0  0  0  0  0]
 [269 270 271 249 272 273 32 274  0  0  0  0  0  0  0  0  0  0  0]
 [ 42 43 44 45 46 47 48 15  0  0  0  0  0  0  0  0  0  0  0]
 [319 702  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0]
 [193 27 13 28  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0]
 [256 427 211  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0]
 [ 84 85 86  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0]
 [ 21 152 430 431 47  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0]
 [110 174 362 40 122  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0]
 [1 0 0]
 [1 0 0]
 [1 0 0]
 [0 1 0]
 [1 0 0]

```

Figure A.3 Evaluation screenshot

Appendix B - Screenshots of Web UI

Figure B.1 shows the main page of Domain classification. It contains a text box for input of Sentence and submit button to get output.

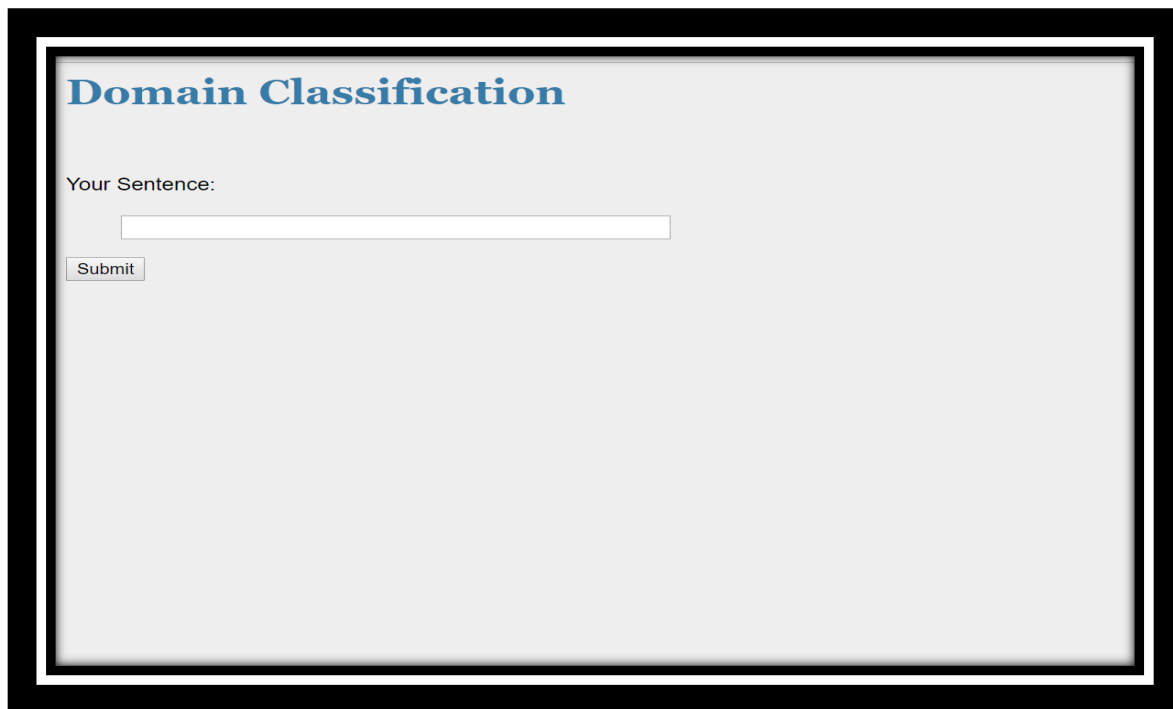
The screenshot shows a web interface titled "Domain Classification" in blue text. Below the title, there is a label "Your Sentence:" followed by a white text input box. Below the input box is a "Submit" button. The entire interface is enclosed in a black rectangular frame.

Figure B.1 Domain Classification Web UI

Figure B.2 shows the output of the sentence “is it drizzling today?” of weather domain on the UI and further input of the domain.

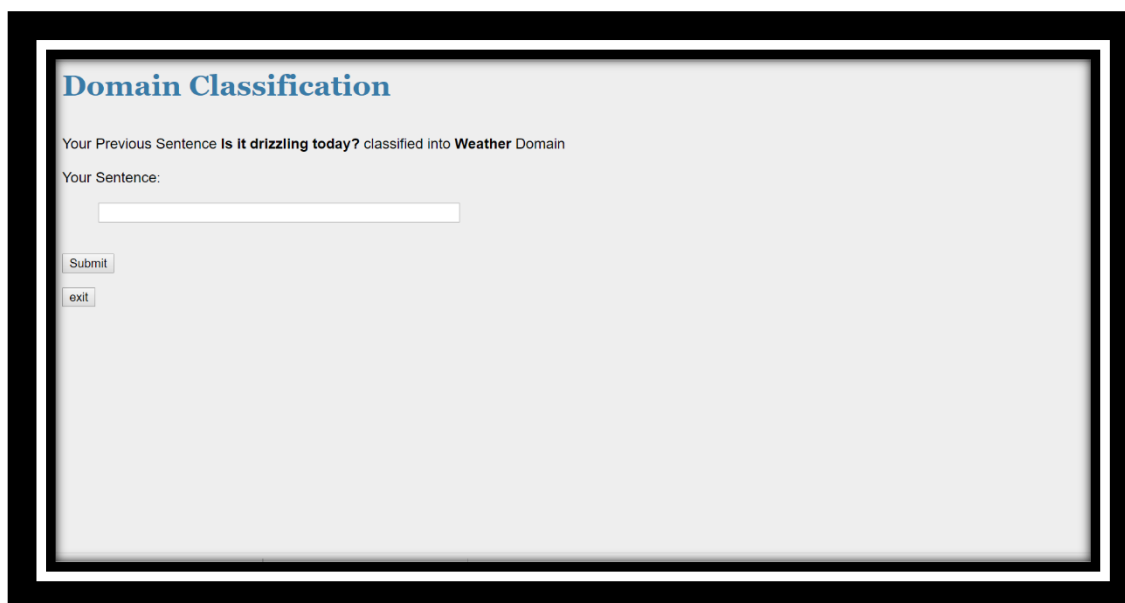
The screenshot shows the same web interface as Figure B.1, but with additional output. Below the "Your Sentence:" label, there is a line of text: "Your Previous Sentence Is it drizzling today? classified into Weather Domain". Below this text is another white text input box. Below the input box are two buttons: "Submit" and "exit". The entire interface is enclosed in a black rectangular frame.

Figure B.2 Output of the Classification

Figure B.3 shows the graph of the accuracy during training and the evaluation. The graph is plotted between the number of sentences and the time. The blue line is during evaluation whereas the orange is the output.

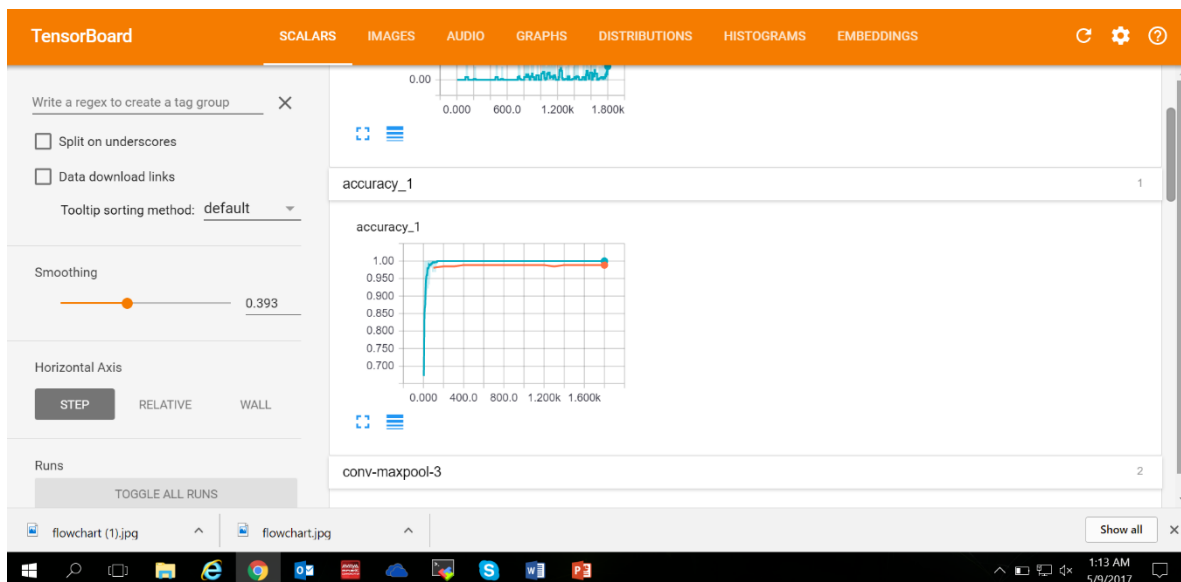


Figure B.3 Accuracy graph through Tensorboard