GENERIC INFORMATION EXTRACTION FRAMEWORK FOR DOCUMENT PROCESSING

Agampodi Kanishka Gayathri Silva

189395H

Thesis submitted in partial fulfilment of the requirements for the degree of Master of Science in Artificial Intelligence

Department of Computational Mathematics

University of Moratuwa

Sri Lanka

May 2021

DECLARATION

I declare that this is my own work and this thesis does not incorporate without acknowledgement any material previously submitted for a Degree or Diploma in any other University or institute of higher learning and to the best of my knowledge and belief it does not contain any material previously published or written by another person except where the acknowledgement is made in the text.

Also, I hereby grant to University of Moratuwa the non-exclusive right to reproduce and distribute my thesis, in whole or in part in print, electronic or other medium. I retain the right to use this content in whole or part in future works (such as articles or books).

Name of the Student

A.K.G. Silva

Signature of the Student

Date: 31/5/2021

Supervised By

Dr A.T.P. Silva

Signature of the Supervisor

Date: 31/5/2021

DEDICATION

I dedicate this thesis to my family and friends. I will always appreciate the help of friends for the things all they have done and their valuable thoughts. I dedicate this work and give many thanks to people at the University of Moratuwa for their help and especially for lecturer for guiding this research.

ACKNOWLEDGEMENT

First and foremost, I would like to extend the sincere gratitude to my supervisor Dr A.T.P. Silva, for the continuous support of the research, for the patience, motivation, enthusiasm, and immense knowledge. I would like to thank the University of Moratuwa for allowing carrying out this research and it continuing support during the research.

I would like to pay gratitude for all the academic and non-academic staff members of the University of Moratuwa and the batchmates for their generous support, comments and encouragement throughout the project. I am grateful to all expert and novice meditators who were involved in this research project.

ABSTRACT

Information extraction from documents has become great use of novel natural language processing areas. Most of the entity extraction methodologies are variant in a context such as medical area, financial area, also come even limited to the given language. Rather than tackling this problem in such manner, it is better to have one generic approach which is applicable for any of such document types to extract entity information regardless of language, context and structure. Also, the great barrier in such research is exploring the structure while keeping the hierarchical, semantic and heuristic features. Another problem identified is that usually, it requires a massive training corpus. Therefore, this research focus on mitigating such problems.

Throughout the research timeline, several approaches have been identifying towards building document information extractors focusing on different disciplines. Starting from optical character recognition of document images to data mining of large corpus of documents this research area has been contributed to the development of natural language processing, semantic analysis, information extraction and conceptual modelling. Although in separate ways those are trying to achieve the generic ability to process any kind of document which unfortunately not being achieved successfully due to the approach and technical limitations.

As per the approach within this research, it can process any kind of document in any domain by simply adhering the conceptual relations without being trying to extract component-wise and mapping into known structures. Just as a human being look at any unknown document and going through the relations and making best guesses on answering the queries, this system will also mimic the same behaviour. As per the output, it can either document Concept-Relation or some answer for the given query.

The experimental strategy has partaken with regards to several different datasets originated from SQUAD 2.0, DOCVQA dataset, SQUAD 2.0 dataset and Kaggle based datasets. Based on F1 evaluation metric it performs with overall 87.01 performance rate on SQUAD 2.0 dataset showcasing its capable of question-answering task with higher accuracy.

Upon diving into experimental design, starting from the dataset evaluation several experiments have been carried out. Datasets such as SQUAD 2.0 and DocVQA has been used to evaluate the overall performance over metrics such as F1 score, accuracy and ANLS providing scores 87.01,52.78 and 0.583 respectively. The F1 score, which is 87.01 showcase that the provided solution achieves the expected objectives in deriving a generic model fitting for any question-answering task based on documents.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

Abbreviation	Description
GIEF	Generic Information Extraction Framework
DLS	Document Concept-Relation
NLP	Natural Language Processing
OCR	Optical Character Recognition

1. INTRODUCTION

1.1 Prolegomena

Digitization of documents and moving forward for the paperless community has been in major discussion in past few decades but also having to encounter difficulties alongside, it has generated many research opportunities advancing from image processing, Natural Language Processing, ontology, semantic analysis, etc. Such researches have usually partaken in addressing a problem, but none or less are supporting to provide a generalized approach for information extraction from documents. Document processing is harder due to its diversity, even in document generation could be in different ways, i.e. handwritten, scanned handwritten, typewritten, machine-generated, machine-printed. In terms of content, again it is possible to divide among purpose (business, legal, technical, medical, academic, historical) [1] [2], language (English, Hindi, Sinhala, Arabic, Multi-lingual) [3].

Purpose of the document processing may concern answering a given query/question, grouping a set of documents as the answer for a query, extract a part of the document (e.g. flowchart, image), converting to the digitized for further storage. To achieve all these, it is better to have a more generalized solution than limited to the scope of each application areas. However, this would not be an easy task due to diversities in document nature, but worth exercising since it would privilege providing a novelty way of addressing document processing issue.

The problem in scope will consider addressing in two phases – querying problem, extraction problem. First, we would consider the querying problem, so that, we should address all the document types i.e. pdf, handwritten hard copies, images, should be multi-lingual, should be able to OCR [4] and convert any complex content including charts, images, diagrams. Then we would investigate the querying problem whereas it could vary in the form of the question in natural language, form to fill using document information, entities to extract.

This research will address a more generalized framework for document information extraction. Here we will consider how humans perceive information from documents even though less experienced with the content and less knowledge on the language by

mapping the information blocks and deriving semantics. One of the drawbacks in existing approaches is those are merely considered on OCR the document first and then trying to convert it. But if we could mimic the way a human extracts such information, for example by deriving relations among information blocks, it would provide a more effective approach. As the initial step, this would be limited for documents but would be extendable towards video processing, voice analysis etc.

In [5] explain a reading strategy of humanized approach of document extraction, where first skim through the test to get a general idea, read the question or query carefully with the equipped general understanding from the document and then search for the answer in the document with the prior knowledge. The proposed method in this paper provides an approach with neural networks.

If we consider the contribution towards the technology, this research will not merely be using the existing researches, tool and libraries. We will divide the whole process into several key areas and will address issues in each area while proposing new algorithms. Such areas would include OCR, semantic analysis, entity recognition and natural language processing.

This will discuss the objectives of the research, some background and motivation by assessing a literature study, the problem, in brief, proposed solution in terms of inputs, outputs and the methodology drafted, resource requirements for the experimental setup, actions plans in a timeline and finally a reference for the research papers cited.

1.2 Aim and Objectives

1.2.1 Aim

Develop an automated generic information extraction framework for document processing

1.2.2 Objectives

- Critically review the literature in current researches and tools for document processing and information extraction
- Design and develop a generic framework for document information extraction

- Implement a generic framework which capable of performing any document processing
- Evaluate the developed generic framework designed for document information extraction

1.3 Background and Motivation

By considering the current state of the art we can observe various attempts to approach document information extraction, but addressing different scopes. Purpose of the document processing may concern answering a given query/question, grouping a set of documents as an answer for a query, extract a part of the document (e.g. flowchart, image), converting to the digitized for further storage. To achieve all these, it is better to have a more generalized solution than limited to the scope of each application areas. However, this would not be an easy task due to diversities in document nature, but worth exercising since it would privilege providing a novelty way of addressing document processing issue.

1.4 Problem in Brief

The problem in brief of the considered research is a concern on providing a generic approach of processing any kind of document and perform the necessary information extraction. Current approaches are limited to a given scope, document type, structure, language etc. Applying these researches in real-time applications will not provide aid for day to day applications. If we are to consider the existing approaches, a large corpus of documents used for training and fitting the model so that the requirement for a more generalized framework which is capable on extracting information from any kind of document exists. The key idea behind this problem is somewhat mimic how a human understands an unknown document and understand through the content and the relation between the sections.

1.5 Generic Information Extraction Framework for Document Processing

This research is subscribed to fulfil 2 promises - addressing the problem statement and contributing to the area of research. With those objectives in mind, we could derive

the study into several disciplines as Image Processing and OCR Solution, formulating Concept-Relation, semantic analysis, NLP, entity recognition.

Image processing and OCR solution will comprise preprocessing, noise removal, character recognition, segmentation. Formulation of the Concept-Relation will consider statistical and structural properties of the document. Semantic analysis to not only deriving the semantic of the text-only but additionally incorporating the geological distribution of the words and phrases. Natural language processing would be focused on formulating the queries to answer as it would come in different forms such as questions, forms, tables, etc. Entity recognition will also to be considered as it is applicable majorly in form information extraction to identify keyword as those would appear in different linguistic variations. In each area, we would address new algorithmic approaches to address the limitations identified.

In terms of input, it would be the document image or system-generated document along with a query statement. The output is the answer for the given query, which could come as an answer to the query, filled form, labelled document which dependable on the input given.

1.6 Resource Requirements

As this research focuses on delivering a generalized solution and not focusing on one limited scope, we will have to adhere to features from the vast diversity of documents. DocVQA dataset would be the starting point and, we would use Kaggle datasets to explore many diversities. If it would be possible to get organizational data such as from universities, hospitals, then the masking would perform due to privacy concerns.

1.7 Structure of the Thesis

The thesis structure would be as follows. In Chapter 2, a critical literature review has done across the related state of the art with the analysis of research gaps and limitations whereas those will be identified to formulate the problem definition. Then in Chapter 3, a short discussion on technology adapted has been taken with relevant references. In Chapter 4, the focusing area is the overall approach for the system considering input, output, process, users and features. Having those items being set, within Chapter 5,

describing the design of the system with regards to submodule architecture. Then the content follows Chapter 6 with regards to the implementation considering algorithms and technical points. Evaluation criteria are being discussed within the Chapter 7 with experimental design and results and the Thesis is concluded with Chapter 8 conclusion. Finally, several Appendices has been included with additional materials such as algorithms, flowcharts and source codes.

1.8 Summary

This section provides the introductory remarks for the thesis by considering the background to the thesis to the proposed solution. Finally, it provides an overview of the structure of the thesis document. Then in the next section, it will discuss the literature study summary on considering the related researches.

2. STATE OF THE ART IN DOCUMENT INFORMATION EXTRACTION

2.1 Introduction

The literature study is carried out scoping different disciplines since it covers every objective as stated. During the research, many other academic materials will be studied as this is the initial setup.

2.2 Major Researches in Document Information Extraction

2.2.1. OCR Handwritten Documents

In the paper [6], it discusses on language-independent rule-based classification for handwritten text and the printed text for data entry forms. This is mostly exercised in extracting information from forms. It uses structural and statistical features derived into a rule-based system to determine the origin of the text. It considers baseline characteristics and stroke characteristics to differentiate.

Chung and Detail [4] propose a methodology for full-page offline text recognition using deep neural networks incorporating with object detection neural networks, multi-scale CNN and bidirectional LSTM to perform text localization and then the text recognition. In-text localization it performs passage identification and line segmentation using IAM datasets and ResNet34 on ImageNet. Then during the text recognition phase, CNN-biLSTM will output NXM in terms of probabilities and with the language modelling, it has converted to a string of text using the beam search approach. The following diagram will illustrate the overview of the system they proposed.

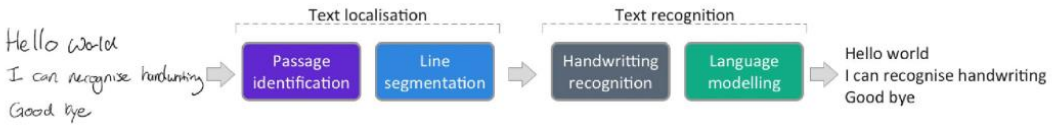


Figure 2.1: Overview of the Text Recognition System *[4]*

2.2.2 Derive Concept-Relation

Segmenting stream of the document using structural and factual descriptions is discussed in [7]. It has introduced 4 document levels such that, records, technical documents, cases and fundamental documents. This proposes a novel method for stream segmentation in stream data. To compare descriptors, a logbook for the previous pages has been used. The system diagram is illustrated as follows.

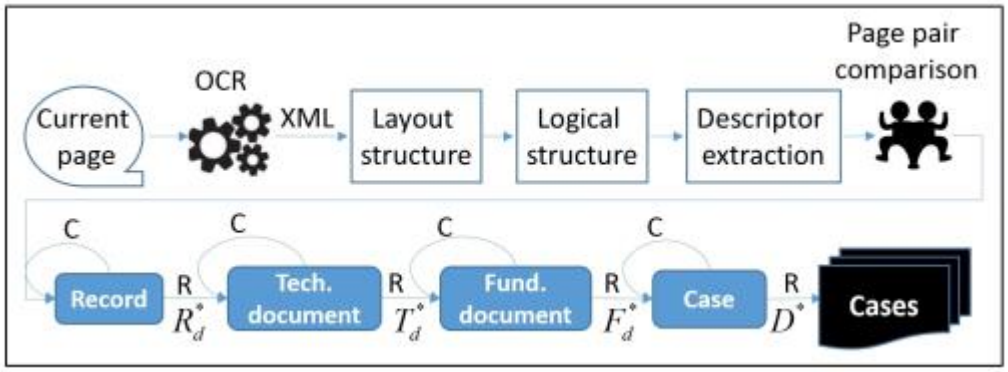


Figure 2.2: System Diagram of Descriptors *[7]*

Logical labelling is been discussed in the study [3], using 2D Conditional Random Fields to derive the page structure in machine-printed documents. It constructs neighbourhood graphs and pairwise clique templated while combining local and contextual features obtained from the PDF document. Using SVM classifier, it derives the baseline without any contextual information considered. In prior knowledge-based context analysis, it considers basic context, the relation between text and text, non-text and text.

One of the OCR application is digitizing historical books in libraries where the searchable text is derived. Although the existing OCR applications extract simple structures such as paragraphs and pages, it is harder to extract complex structures such as chapters, sections. Therefore, mapping such table of content to provide more structural information is proposed in [8]. It introduced aggregation-based method on two set operators and properties of the table of content entries.

2.2.3 Derive Semantic Structure

Document Semantic Structure Extraction is a separate research discipline where document image is segmented into the region of interest and provide a semantic explanation for each [9]. This involves two phases, page segmentation and Concept-Relation analysis. Yang and Yumer [9] presents a multimodal fully convolutional network to extract semantic structure in terms of pixel-wise segmentation task.

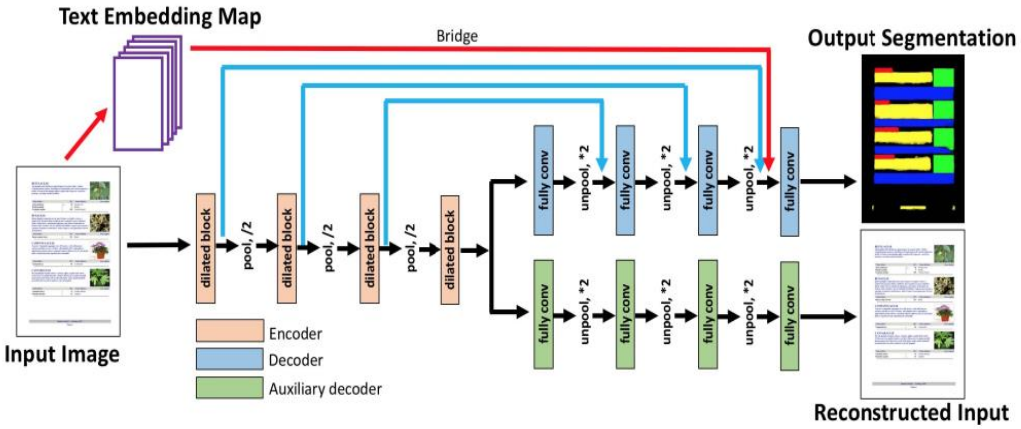


Figure 2.3: Multimodal Fully Convolutional Neural Network *[9]*

The paper [10] propose an approach to dimensionality reduction of feature vector size for web text document classification using WordNet ontology. Also, it uses Vector Space Model for the mining task and for the term weighting it has used Term Frequency Inverse Document Frequency. The suggested approach is ontology base and for the evaluation, it uses Principle Component Analysis.

2.2.4 Document Entity Recognition

Differential tools are evaluated in [11]. In the entity recognition process lexical, semantic and heuristic features are analyzed. English corpus of 579 words has been used. Based on entity types results are evaluated. Finally, this paper proposes a model that eliminates limitations identified. Also, the corresponding model. is based on the small corpus. Best tools are Supersense - WNSS and lowest performance from Afner. In [1], it uses NLP pipeline begins with part of speech tagging and then use chunking to obtain NER. It has taken into consideration of shallow syntax in tweets. For that annotated tweets and build tools trained on data. Also, have used features generated

from T-POS & T-Chunk in segmenting named entity. The paper [12], discusses the meaning behind Named Entity Recognition, and analyze the field from both theoretical and practical point of views considering drawbacks, challenges and opportunities in the area. It brought the argument to show that this task mainly depends on the development and evaluation of the tools. Also, it checks upon further research areas where this area could be further developed.

NER can be applied in many practical scenarios, one is the usage in analyzing Twitter posts. In the paper [13], discusses such usage. It identifies several Named Entities and classifies them among People, Locations and Organizations. Also, it brought into the discussion about identifying the language-specific features and methods effectively so that this research can be further advanced into the areas of NER performing on formal sources such as news articles and less structured microblogging texts. Throughout the process of research, it identifies the difference between the microblogging text and the formal proses.

The paper [14], proposes a framework for named entity detection from Web content associated with semi-structured text data records, by exploiting the inherent structure via a transformation process facilitating collective detection. To learn the sequential classification model, this framework does not require training labels on the data records. Instead, it makes use of existing named entity repositories such as DBpedia. It incorporates this external clue via distant supervision, by making use of the Generalized Expectation constraint. Finally, a collective detection model supporting the logical inference is proposed to consider the consistency among potential named entities and also header text.

2.2.5 Question Answering from Documents

Since this framework should be able to provide answers for the given queries, we have to consider the design aspects of question answering systems, where it requires the composing of several components, Named Entity Recognition, Semantic Analysis of the question, Named Entity Disambiguation, query expansion and execution, etc. In the study [15], it has proposed, Web And semantic knowledge Driven (WAD) approach to increase the accuracy of the document selection. This approach first

evaluates the user query, entity linking method and the query expansion technique. Finally, ontological information is used to rank the answers.

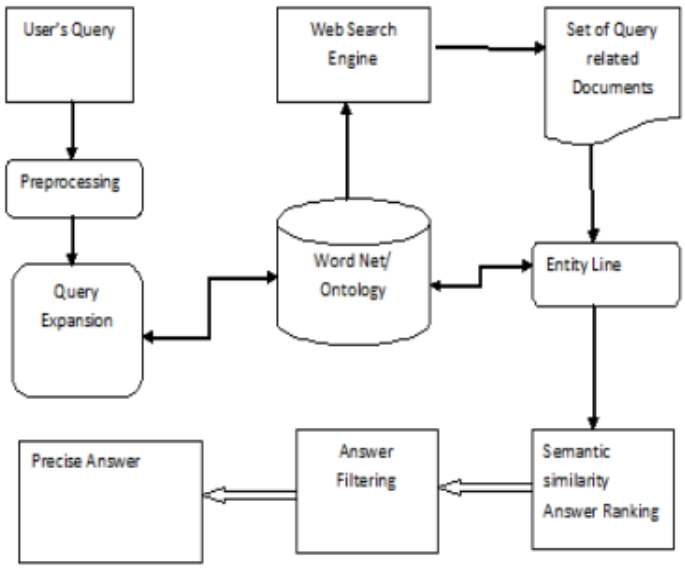


Figure 2.4: Process of WAD Methodology *[15]*

In the paper [16], it suggests a graph-based Question Answering system for reading comprehension tests, where it picks the sentence in the given passage which gives the best answer. It uses combined techniques of morphological analysis, coreference resolution and synonym checks to achieve higher accuracy.

Table 2.1: Summary of the Literature Study

Research	**Problem Addressed**	**Technology**
[1] Language-Independent Rule-Based Classification of Printed & Handwritten Text (Classification of Printed & Handwritten Text)	Categorize Source of Document Content – Handwritten/Printed	• Rule-Based • Baseline characteristics • Stroke characteristics
[2] A Computationally Efficient Pipeline Approach to Full Page Offline Handwritten Text Recognition	Full page offline text recognition	• Deep Neural Networks • Object Detection Neural Networks

		• Multiscale CNN • Bidirectional LSTM
[3] Combination of Structural and Factual Descriptors for Document Stream Segmentation	Segmenting stream of the document using structural and factual descriptions	• logbook for the previous pages • stream segmentation
[4] Logical Labeling of Fixed Layout PDF Documents Using Multiple Contexts	Derive the page structure in machine-printed documents	• 2D Conditional Random Fields • SVM classifier
[5] Enhancing Table of Contents Extraction by System Aggregation	Digitizing historical books in libraries where the searchable text is derived	• Aggregation-based method on two set operators
[6] Learning to Extract Semantic Structure from Documents Using Multimodal Fully Convolutional Neural Networks,	Extract semantic structure in terms of pixel-wise segmentation task	• Multimodal fully convolutional network
[7] A Novel Approach for Ontology-based Dimensionality Reduction for Web Text Document Classification	Web text document classification	• Dimensionality reduction of the feature vector • WordNet ontology • Vector Space Model for the mining task • Term Frequency Inverse Document Frequency for term weighting
[8] Evaluation of Named Entity Extraction Systems	Evaluate current literature	•
[9] Named Entity Recognition in Tweets: An Experimental Study	Segment Tweets	• Speech Tagging and Chunking

		• T-POS & T-Chunk
[10] Named Entity Recognition: Fallacies, Challenges and Opportunities	Evaluate current literature	•
[11] Named entity recognition: adapting to microblogging	Identifies several Named Entities and classify	• Language-specific features
[12] A proposal to automatically build and maintain gazetteers for Named Entity Recognition by using Wikipedia	Named entity detection from Web content associated with semi-structured text data records	• Exploiting the inherent structure
[13] Automated Question Answering System Using Ontology and Semantic Role	Increase the accuracy of the document selection	• Web And semantic knowledge Driven (WAD) approach
[14] A Graph-Based Relation Extraction Method for Question Answering System	Reading comprehensive texts	• Morphological analysis • Coreference resolution • Synonym checks
[16] Document Structure and Layout Analysis	Extract Physical and Concept-Relation	• Top-Down and Bottom-Up Approaches • Grammar-like method • Docstrum and Voronoi Algorithms
[17] A knowledge-driven approach to biomedical document conceptualization	Cluster documents based on ontology (user-specified)	• Corpus related ontology generation algorithm • MeSH, GO ontologies

[18] A Concept-Driven Automatic Ontology Generation Approach for Conceptualization of Document Corpora	Identify the key concepts and automatically generate ontologies based on these concepts for the conceptualization of document corpora	• Semantic relationships in corpus • Clonto method • STC Algorithm • Fuzzy Ants clustering algo
[19] A Parallel Neuromorphic Text Recognition System and Its Implementation on a Heterogeneous High-Performance Computing Cluster	HPC-based context-aware intelligent text recognition system (ITRS) that serves as the physical layer of the machine reading	• Word and Sentence Confabulations • BSB Model – Brain-State-in-a Box • Cogent Confabulation
[20] Accelerating Pattern Matching in Neuromorphic Text Recognition System Using Intel Xeon Phi Coprocessor	optimized implementation of Brain-State-in-a-Box (BSB) neural network model on the Xeon Phi coprocessor for pattern matching in the context of intelligent text recognition of noisy document images	• BSB Model • Cogent Confabulation

2.3 Limitation in Current Approaches

The overall summary of this literature study is illustrated in Table 2.1. Most of the aforementioned researches are firmly depend upon the dataset/corpus and also limited to the addressed language. The suggested model usually trained and implemented within the given environment whereas less performance for completely strange unstructured documents. For some researches, a precompiled training dataset is required to build the model, so that when applying within a different environment an additional effort need to be paid off for preparing such datasets. Also, the current state

of art referring to this study usually subjective for the problem area such as academic, business, bio-medical etc.

2.4 Future Trends

In terms of future trend regarding this research area, it is towards the Neuromorphic Information Processing. Also, rather than being limited to one linguistic area or subject scope, most recent studies are moving towards extracting information by considering the conceptual factors rather than digitizing the text and then grabbing the meaning of the document.

2.5 Problem in Brief

Current approaches are limited to a given scope, document type, structure, language etc. Applying these researches in real-time applications will not provide aid for day to day applications. If we are to consider the existing approaches, a large corpus of documents used for training and fitting the model so that the requirement for a more generalized framework which is capable on extracting information from any kind of document exists. The key idea behind this problem is somewhat mimic how a human understands an unknown document and understand through the content and the relation between the sections.

2.6 Summary

This chapter described the literature review representing each module. Next chapter will be about Technology Adapted for the development of the research.

3. THEORETICAL FOUNDATION FOR GIEF

3.1 Introduction

This chapter will mainly focus on the theoretical foundation for the research, which will be used within the phases of design and implementation and even for the evaluation purpose. The proposed framework is being inspired by Transformers, BERT [17], ELMo [18] architectures while using TensorFlow as the base for implementation. The following sections will thoroughly analyze the inspiration points along with a critical analysis of the reasons to select the given theoretical concepts over other bases.

3.2 TensorFlow

TensorFlow is designed to provide an API for neural network modelling. It is known as a general-purpose open-source library that is open-sourced by the Google Brain team and licensed under Apache License 2.0. The current version is known as TensorFlow 2.0 while previously it has followed a series of releases namely TF 1.0.

TensorFlow can be used for various tasks but it has gained recognition in modelling deep neural networks. TensorFlow can run on multiple GPUs and CPUs along with extensions for GPUs.

It uses stateful dataflow graphs for the computations. It uses tensors, which are multidimensional data arrays where computations occur.

3.2.1 Tensor Processing Unit (TPU) [19]

Tensor Processing Unit (TPU) is specifically built for machine learning. It is a hardware chip in the category of application-specific integrated circuit (ASIC). This is specifically tailored for TensorFlow applications. TPU is a programmable accelerator and it mainly focuses on using and running models. In the 2nd generation of TPU, it is well known with Google Compute Engine while delivering 180 teraflops of performance. During the 3rd generation, its performance has been raised to 420 teraflops with high bandwidth memory of 128 GB.

Increasing batch size is a way of achieving further accelerations. By convention, 128 elements are used per core, which results altogether 1024 batch size. When it is being increased, TPU will crunch data faster. That is why we usually increase the learning rate alongside batch size.

To mitigate data bottleneck situations, several strategies are in place. When this occurs, TPU is idling while waiting for data. When TPU is reading data from google cloud storage, it can sustain a large throughput by continuously streaming multiple files simultaneously. Usually, TFRecords is used as a container format for data being used in TPU, where the following code block defines a reading dataset from TFRecords files, Parallel streaming can also be enabled. Namely, there are 2 settings for this purpose.

1. num_parallel_reads = AUTO: Read multiple files only if available.
2. experimental_deterministic = FALSE: Data order enforcement is disabled as it will require to shuffle data. This enables streaming any TFRecord faster.

3.2.2 Features [20]

It provides Python and C APIs and also including several other 3rd party libraries for MATLAB, R, and Scala. The main features in TensorFlow include flexible, responsive construct, easily trainable, open-source, large community, etc. It facilitates a visualizer with TensorBoard which can be used to inspect model representations.

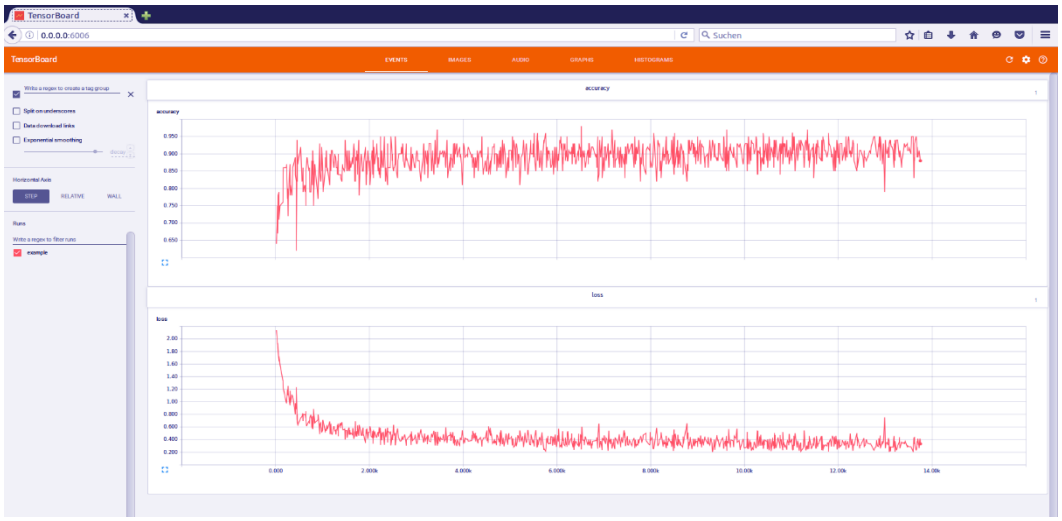


Figure 3.1: TensorFlow Visualizer with TensorBoard *[20]*

Also, TensorFlow facilitates Layout Components with functions such as tf.contrib.layers. It provides layered operations related to weights and also enables the convolution layer, batch normalization, and dropout layer. Layers can be optimized with tf.contrib.layers.optimizers such as SGD, Adagard, Moment.

With the use of distribution functions, it helps with Bayesian like models with Bernoulli, Chi2, Gamma functions. Also, TensorFlow has got feature columns, which is an intermediate between estimators and raw data. Feature column implementation steps are defined in Figure 3.2

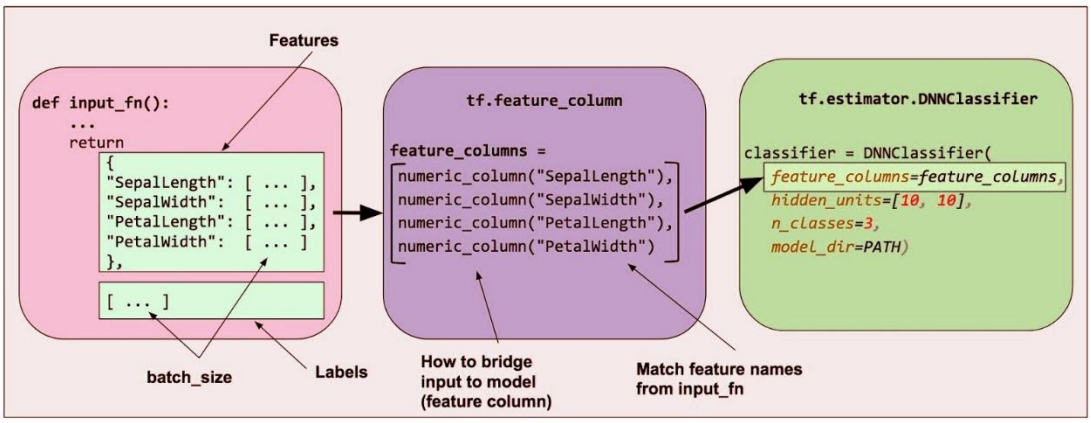


Figure 3.2: TensorFlow Feature Columns *[20]*

Another well-known feature in TensorFlow is that it is Open Source. Therefore, it is a community built. TensorFlow provides pipelining whereas it enabled training multiple GPU and multiple neural networks. With this feature, TensorFlow models are very efficient in large-scale systems.

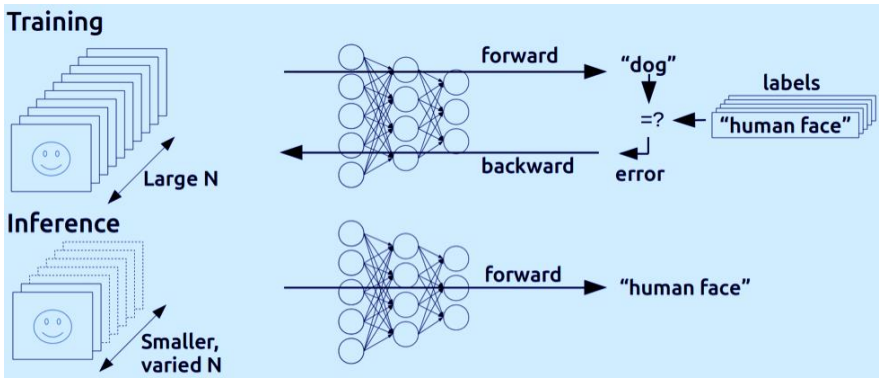


Figure 3.3: TensorFlow: Parallel Neural Network Training *[20]*

3.2.3 Applications [21]

Several well-known applications using TensorFlow includes RankBrain, Collaboratory. Google Colab provides a TensorFlow Jupyter notebook environment for processing. RankBrain is a large-scale deep neural net which used for search ranking algorithm in Google to sort billions of pages and provide the most relevant. Inception Image Classifier provides a baseline model for computer vision. Also, another application of TensorFlow is Massive Multitask which is from Stanford University for Drug Discovery using deep neural network model.

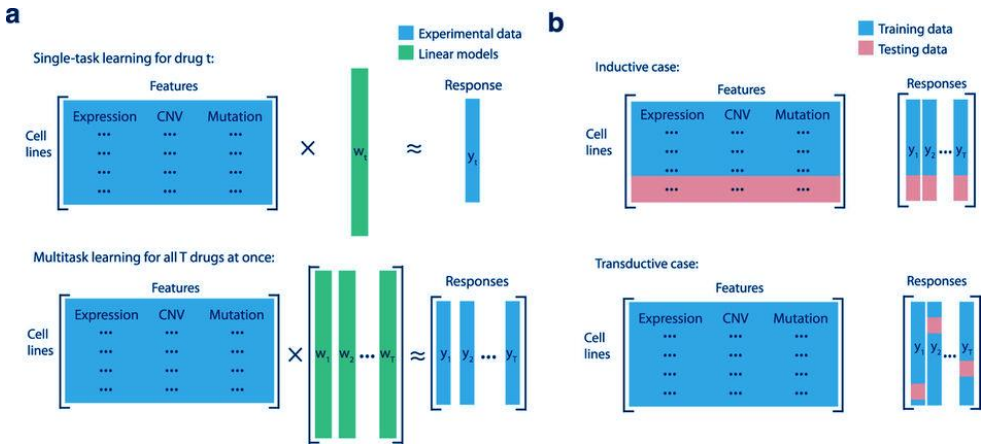


Figure 3.4: Massive Multitask *[21]*

Also, TensorFlow provides an easy API to predict and train which in line with machine learning tools in Python. Apart from these, other TensorFlow applications include self-driving cars, text summarization, image/video recognition and tagging, speech recognition system and sentiment analysis.

3.2.4 TensorFlow 2.0 [22]

TensorFlow 2 mainly focuses on providing a static computation graph with Chainer and PyTorch. It is a community-driven easy-to-use platform. It provides a more enhanced ecosystem targeting not only researchers but also enterprises. TensorFlow 2.0 provide inheritable interfaces such as variables and checkpoints, which greatly provide reusability.

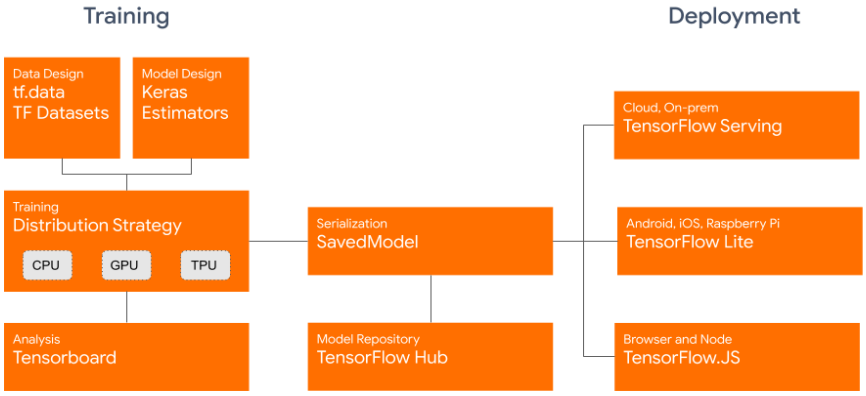


Figure 3.5: TensorFlow 2.0 Overview *[22]*

The standardized format, SavedModel allows us to run models on different platforms such as web browser, cloud, mobile, and even on embedded systems, with TensorFlow Server, TensorFlow JS, and TensorFlow Lite. By using the distribution strategy API, it can address high-performance scenarios that also support Keras model.fit and custom loops.TensorFlow datasets provide an interface for different data types such as text, images, and videos. Also, it supports the session-based programming model. tf.functions can be used to convert the code into graphs with the use of AutoGraph.

3.1.5 Why TensorFlow for GIEF?

TensorFlow provides an entire ecosystem with easy model building. It provides multiple abstraction levels suits for any use, also it builds models using high-level Keras API. Using eager execution, it facilitates immediate iteration and also debugging functionalities. The Distribution Strategy API allows it to be trained on different hardware configurations. Since this research carried on cloud-hosted TPUs, sometimes due to resource limitations it might even result in switching into CPU mode from TPU mode. But since Distribution Strategy API definitions don't require changing model definition in each time of changing hardware platform, it has been a greater advantage.

Also, TensorFlow provides powerful experiment environment for research. It can be used to build and train known state-of-art models while still keeping the performance. Using the TensorFlow doesn't mean that fully abandoning Ketas API instead it

providing access to Keras Function API. Since it is open sources, it has built a powerful ecosystem of add-on libraries such as Ragged Tensors, BERT [17], Tensor2Tensor and TensorFlow Probability. Since this research focus on BERT [17] like models, using TensorFlow is the best choice.

3.2 Transformers

Transformers is mainly used in natural language processing for tasks such as text summarization and translations. The main advantage in transformers is it doesn't require sequential data to be inserted in an orderly manner. Due to this, it allows parallelization compared to Recurrent Neural Networks, and also reduce the overall training time due to that. Pretrained models such as BERT (Bidirectional Encoder Representations from Transformers) [17] are using this model. Transformers follows encoder-decoder architecture where the encoder contains several encoding layers and decoder contains several decoder layers. It was designed based on how long sentences are memorized in neural machine translation [20]. If we take an example of how a sentence being processed in this architecture, first it is embedded and then passed into a stack of encoders. The output from the final encoder is passed to each decoder layer.

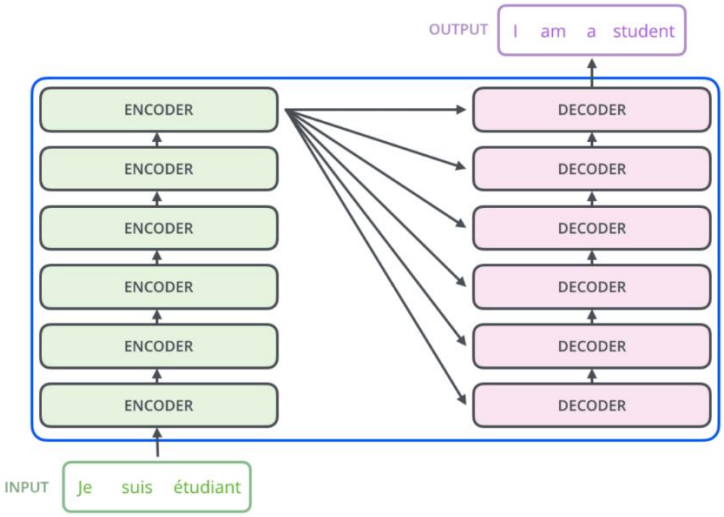


Figure 3.6: Illustrated Transformer *[23]*

3.3.1 Encoder

There are 2 main parts in the encoder, namely, self-attention and feed-forward neural network as illustrated below.

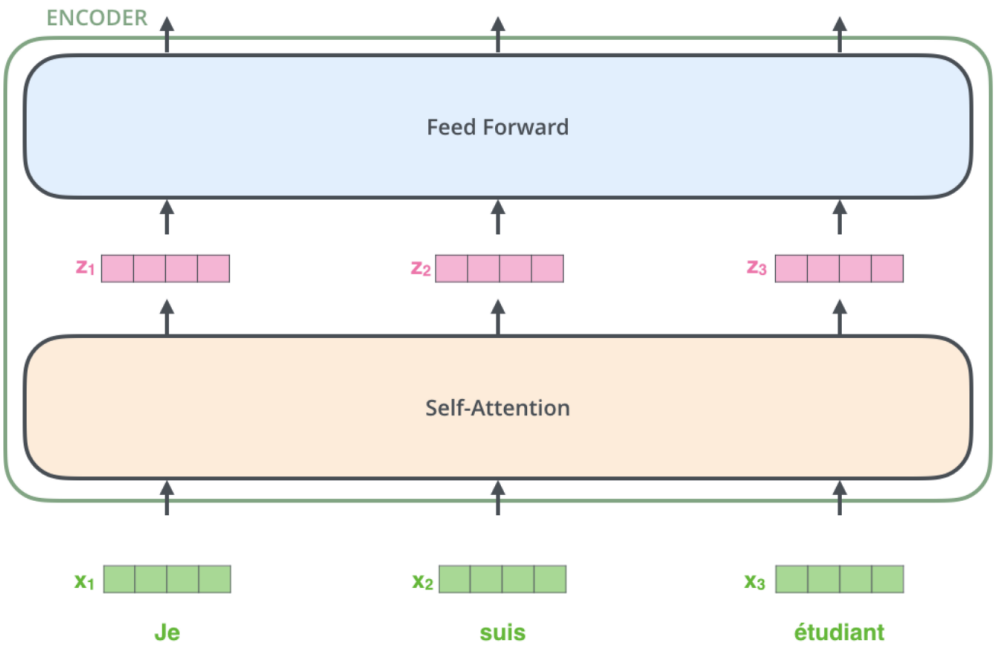


Figure 3.7: Encoder Architecture *[23]*

The sentence is first given to self-attention layer, where it scans every other word and encode to a specific word. For example, each input word x will generate a vector Z through the self-attention layer which takes the input words as (x1, x2, x3, … xn). Then the output of self-attention layer is taken into the feed-forward network which generates output for each input Z. Then this output is fed into the next encoder layer's self-attention block. Likewise, this process goes on through each encoder layers.

In NLP data is in the form of a corpus of sentences, but it needs to be converted into some machine-readable format such as number so that matrix operations can be performed. To convert the given sentence into numbers, word embedding is used. Each word in a sentence is converted to multi-dimensional vectors, usually, this follows GloVe embedding representation. Once the embedding for each input word has been formed, this is passed to the self-attention layer.

There are 4 main learning parameters in the self-attention layer, which are Query matrix (Wq), Key matrix (Wk), Value matrix (Wv) and Output matrix (Wo). Except for Output matrix, other 3 are serving for a special purpose which used to generate special parameters respectively Query(Q), Key(K) and Value(V). The input tensor is

in n x m dimension with n number of input words and m of vector size for each word. Also, output tensors, Q, K, V, Z are having n x dk dimension with n number of input words and usually, dk is 64, which was set by founders of this architecture. Self-attention layer will calculate a vector for each given input word which is called scores.

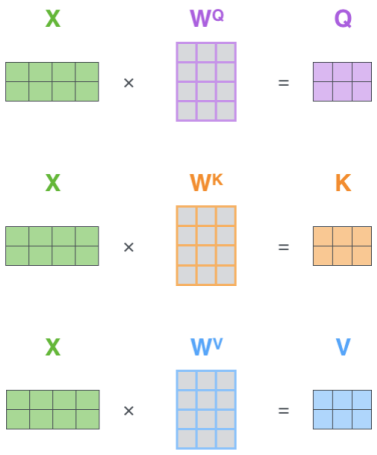


Figure 3.8: Training parameters of self-attention layer *[23]*

Then the vector score calculation occurs for each input word, which is in size of n. This defines how much the selected word gets the contribution from other words.

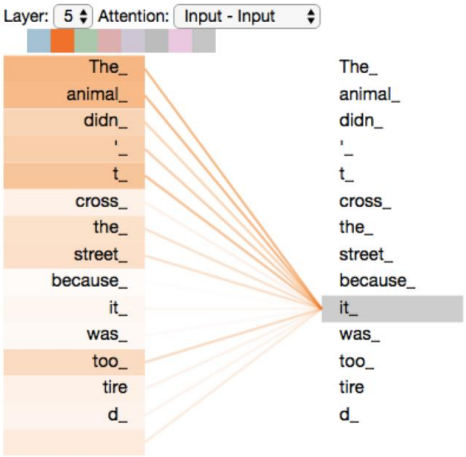


Figure 3.9: Dot product attention

To generate the score vector, it takes the dot product between Q and K vectors. Then the output is normalized by dividing by dk. This result is encoded by using SoftMax function where output is limited unto 1. The SoftMax score elements will be multiplied

with each value vector V, then the output Z of self-attention is obtained by adding all n value vectors.

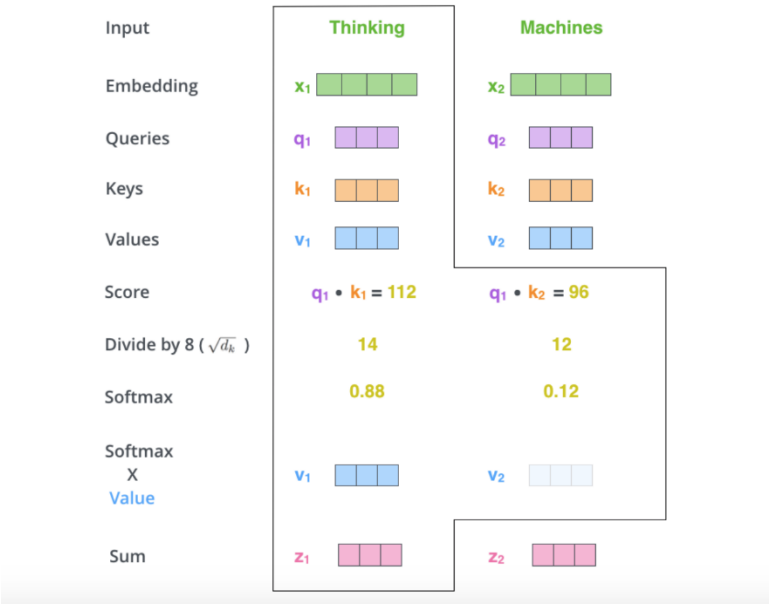


Figure 3.10: Self-Attention Steps Summary *[23]*

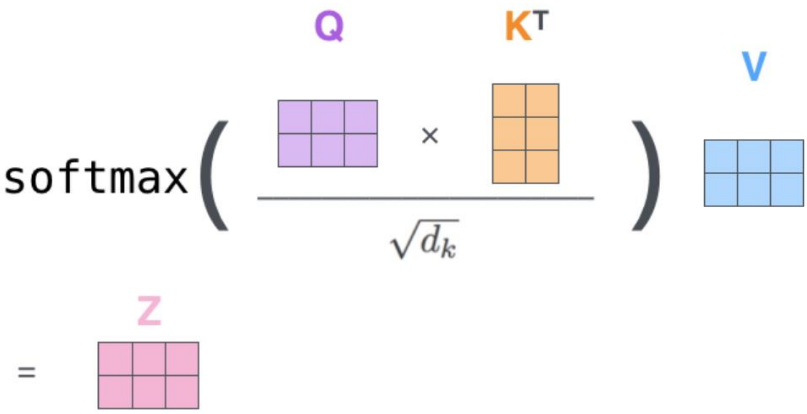


Figure 3.11: Self-attention calculation in matrix form *[23]*

Multi-attention heads are used to reduce any error can occur by single attention head. Following Figure 3.12 explains this process. Within multi-head attention, it uses 8 attention heads.

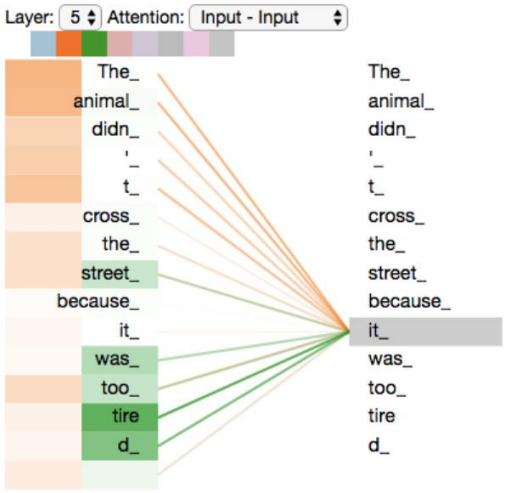


Figure 3.12: Multi Attention Heads *[23]*

Following diagram summarizes all the steps within self-attention layer.

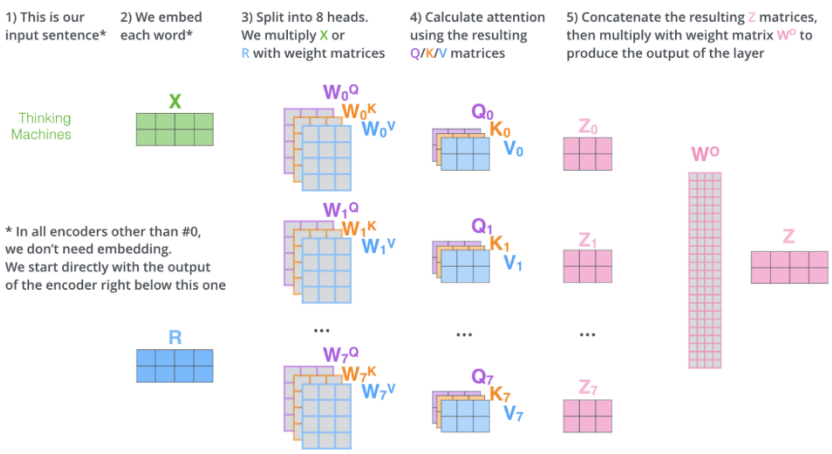


Figure 3.13: Self Attention steps *[23]*

Positional encoding considers where the input word position is, by giving a sense of position to the embedding. Among several ways of positional encoding, it can use

either one-hot encoded vectors or else binary encoding.

$$\overrightarrow{p_t}^{(i)} = f(t)^{(i)} := \begin{cases} \sin(\omega_k . t), & \text{if } i = 2k \\ \cos(\omega_k . t), & \text{if } i = 2k + 1 \end{cases}$$

where

$$\omega_k = \frac{1}{10000^{2k/d}}$$

Following is the result of 128-positional encoding taken from a sentence with a length of around 50 words.

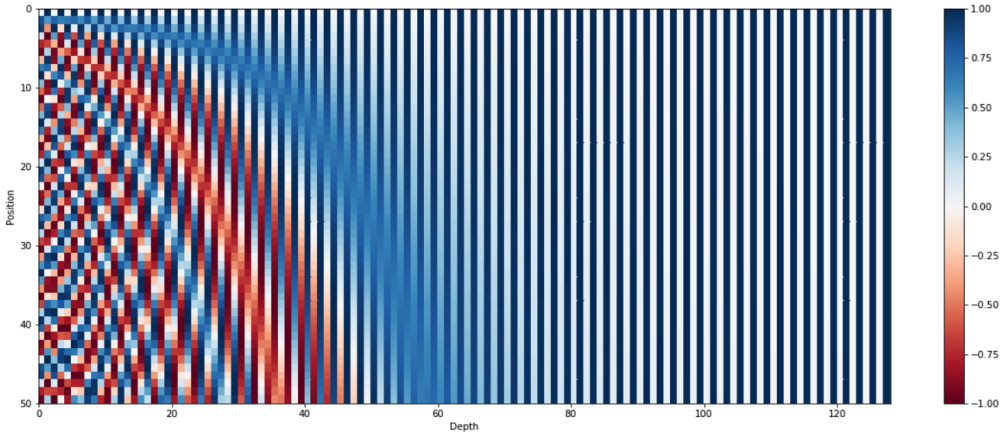


Figure 3.14: Position Encoding Example *[24]*

3.3.2 Decoder

Decoder follows similar architecture to an encoder which also comprised of self-attention and feed-forward network. There is an Encoder-Decoder Attention in between these 2. It follows the same pattern as multiheaded self-attention.

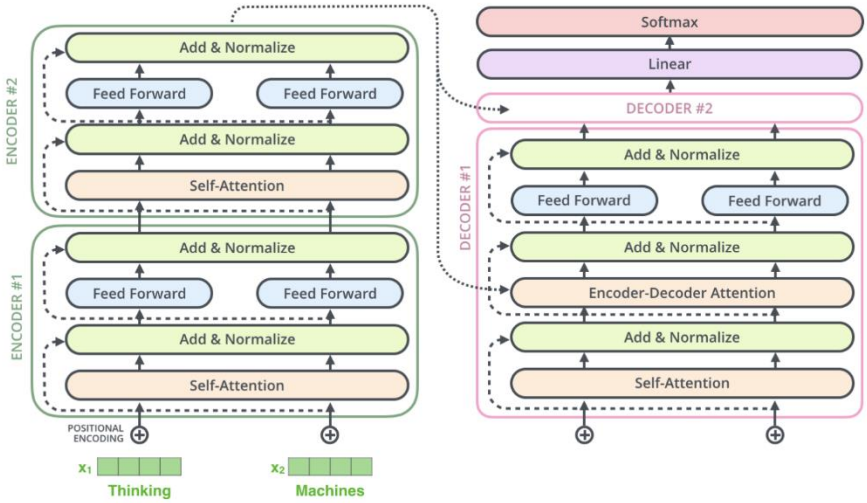


Figure 3.15: Encoder-Decoder Architecture *[23]*

It takes sequential input for the decoder. The output is then passed to another linear layer with SoftMax activations. It follows the same concepts as forwarding propagation and loss functions such as cross-entropy to update trainable parameters.

3.3.3 Why Transformers for GIEF?

This research mainly focuses on deriving concept relations from documents to generate pre-trained models. Approaches such as RNNS, LSTMs will make it a bit difficult to parallelize the input sentence, since those are mainly focusing on sequential inputs and need to provide the input words one by one. But with using transformers, it can capture long-range dependencies as well since the idea here is to consider the given input sequence vector at once. Also, transformers have got self-attention units which are to compute similarity scores between the words in sentences. Transformers avoids recursion due to processing as a whole and learning relationships using multi-attention and positional embeddings. This is the very reason that it is mostly suitable for this research.

3.4 ELMo [25] [26] [18]

ELMo provides deep contextualized word representation using bi-directional LSTM model to get word representations. It analyzes the words by considering the context it's been used. As it uses character-based representation, it creates dynamic vectors from the given input text.

In ELMo [18] it learns both word and linguistic context. The pretrained model can be used for any downstream required NLP task, which are:

1. Question Answering
2. Coreference Resolution
3. Textual Entailment
4. Named Entity Extraction
5. Semantic Role Labeling
6. Sentiment Analysis

3.4.1 Architecture

Multiple word embeddings are produced for every single word considering different scenarios. To capture context-dependent aspects, higher-level layers are used and the model aspects such as syntax are captured by lower-level layers.

$$\begin{aligned} R_k &= \{\mathbf{x}_k^{LM}, \overrightarrow{\mathbf{h}}_{k,j}^{LM}, \overleftarrow{\mathbf{h}}_{k,j}^{LM} \mid j = 1, \ldots, L\} \\ &= \{\mathbf{h}_{k,j}^{LM} \mid j = 0, \ldots, L\}, \end{aligned}$$

x: token representation (word embeddings or character embeddings)
k: position of token
j: #j layer of biLSTM
h: refer to biLSTM layer (First one is forward layer while second one is backward layer)

The ELMo [18] vector and the token embeddings will concatenate to provide a new embedding as illustrated in follows.

$$\tilde{w} = [\,\mathbf{ELMo}_k^{task} \text{ ; Embeddings(w) }]$$

ELMo [18] performs Language Modeling by training to predict the next word for the given word sequence. Not only the next word, it considers the previous word as well. This contextual embedding is by grouping hidden states by concatenation.

There are several approaches to use ELMo pre-trained model, they are:

1. Weighted sum of the 3 layers with word embeddings
2. Fixed mean-pooling without word embeddings
3. Weighted sum of the 3 layers without word embeddings

3.4.2 Why ELMo for GIEF?

One of the requirements in GIEF for document processing is that different concepts can be grouped together and by the same time, each concept groups will have relations at a different level. To facilitate this functionality, context embeddings in ELMo [18] has been used with BERT [17] model. Concept group modelling has been inherited from ELMo [18] architecture.

3.5 BERT [27]

BERT [17] was published by Google AI Language and it uses attention approach by using input words sequence simultaneously to provide contextual meaning. BERT [17] architecture uses transformers base, it consists of several encoder cells just as in transformers. Compared to transformers which use decoder cells to provide an output, in BERT [17] it only consists of encoder part. Since it reads the input words at once simultaneously it is said to be bi-directional.

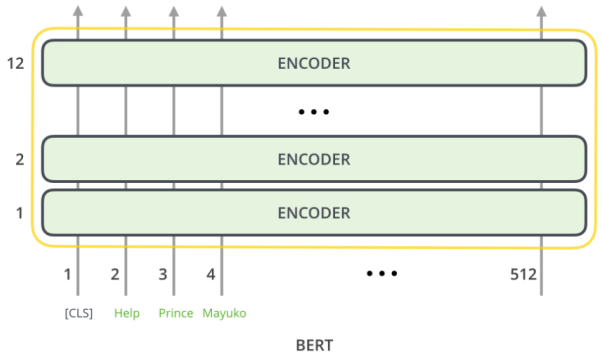


Figure 3.16: Model Inputs in BERT *[28]*

3.5.1 BERT Encoder

Encoder part in BERT [17] is just the same as the architecture in transformers as it consists of feed-forward neural network and self-attention heads. These two segments at the same time act as the training parameters to differentiate within different BERT [17] models.

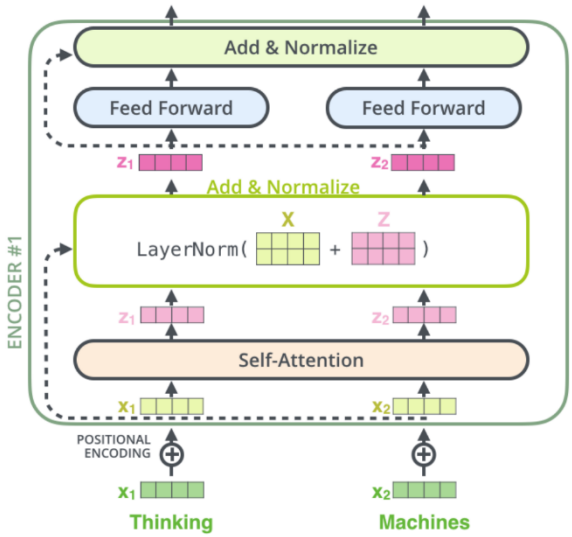


Figure 3.17: BERT Encoder *[28]*

Main two architectures in BERT [17] are BERT base and BERT large, both are taking the input of 512 dimensions. BERT base has 12 transformer blocks with 12 attention heads which resulting 110 million parameters. Its output size is 768 dimensions. In BERT Large it has 24 transformer blocks with 16 attention heads which resulting 340 million parameters where the output size is 1024 dimensions.

3.5.2 Training the Model

BERT [17] is pre-trained on a large corpus of 2500 million words in Wikipedia and 800 million words in book corpus. Masked Language Model (MLM) and Next Sentence Prediction (NSP) are the training methods followed in BERT [17]. In Masked Language Model it's trained by feeding a sentence by masking 15% of words. Using this it makes the BERT [17] predict the masked word using the context of unmasked words. A fully connected layer which uses GELU activation function is fed into the output layer which converts to the vocabulary and applies SoftMax function for the

prediction. To calculate the loss function it only uses the masked word prediction values. This makes the learning much pointed towards context-based.

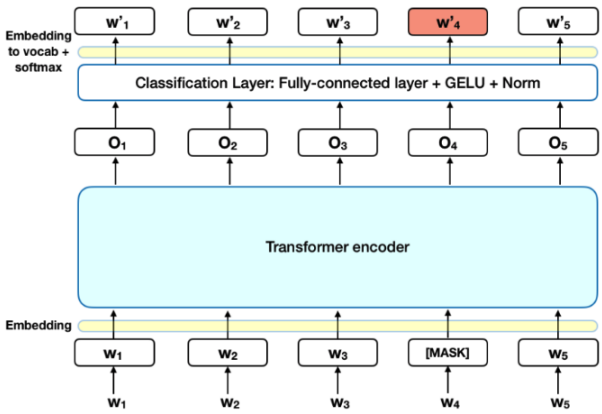


Figure 3.18: Masked Language Model (MLM) *[27]*

During the Next Sentence Prediction (NSP) model is given the input in 2 sentences. Then the requirement is to predict the order of the given 2 sentences. It will return 1 if the first sentence comes after the second sentence during the training and vice versa. The same concept is used in fine-tuning question answering models in BERT [17].

3.5.3 Inputs and Outputs

There are special tokens which used with the BERT [17] model inputs, such as [CLS] and [SEP]. The token [CLS] is added as the first token to denote the start of an input sentence which [SEP] token separate different sentences for example within Next Sentence Prediction phase. Following Figure 3.19 explains a sample input provided to BERT [17].

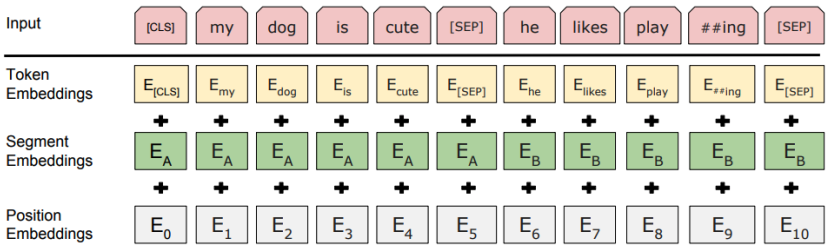


Figure 3.19: BERT *[29]*

Following are the input types in BERT:

1. Token Embeddings
2. Segment Embeddings
3. Mask Tokens
4. Position Embeddings

In Token Embeddings, it represents the input sentence in a numerical representation which segment complex words into tokens. Within Segment Embeddings whenever multiple sentences provided as input, which uses to differentiate sentences. It uses [CLS] and [SEP] tokens for this purpose. Segment embedding will look like in numerical representation such as [0,0,0,0,0,0,1,1,1,1,1] where 0 denotes the elements in first sentence and 1 denotes the elements in second sentence.

Sometimes not all input words are important, then the Mask Tokens are used to mark which words are being padded. Then in the Position Embedding its concern on the input words order.

In usual applications, the output of the BERT will be provided into a feed-forward network with SoftMax as an activation function.

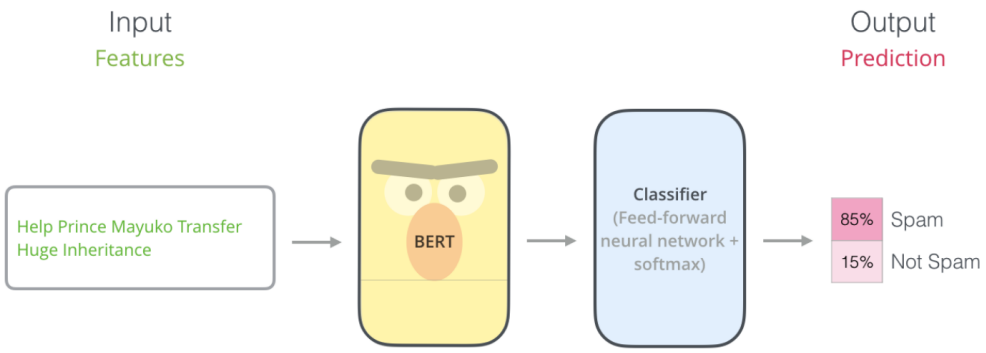


Figure 3.20: Spam detention with BERT *[28]*

3.5.4 Applications

There are different applications of BERT as illustrated in Figure 3.21 which are question-answer modelling, classification task such as for spam detection, answer generation from a given passage, sentence tagging such as entity recognition.

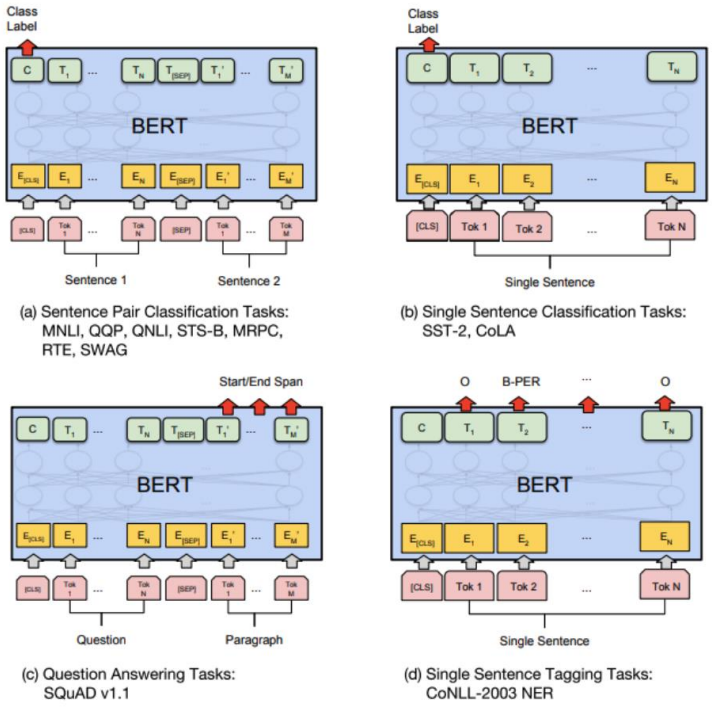


Figure 3.21: BERT Usability *[28]*

3.5.5 Why BERT for GIEF?

Due to the usage of bidirectional flows with masked language modelling, it generates more accurate word representations in deriving conceptual information. In BERT it uses word-level tokenization which is performing better compared to the character level approaches. As this research use word-level embeddings in different levels of abstractions to derive concept relation mappings, BERT models are most suitable. Also, the pre-training and finetuning process for down streaming tasks have been used within this study.

3.6 Summary

This chapter provides a critical analysis of the theoretical background for this thesis with justifications of using particular technological concepts. The discussed theories will be used to design and implement the proposed solution as well as within the evaluation criteria. Next chapter will discuss the researched approach for the given problem statement.

4. CONCEPT RELATION EXTRACTION APPROACH FOR DOCUMENT PROCESSING

4.1 Introduction

To provide a generic approach for document information extraction, concept relation extraction approach has been the most promising methodology, as per the literature study and technical analysis. Overall system approach will be discussed within this section in terms of input, output, process, users and features.

4.2 Hypothesis

The hypothesis behind this research is that using conceptualization it is possible to extract information from documents without limiting to linguistics and scope. The inspiration behind this hypothesis is coming from the observation that how humans trying to understand completely unknown document without having any prior understanding of the language or document content.

4.3 Input

Any type of document which can be either handwritten document, printed document, document images is primary inputs for the system, which will be subject to the processing. The secondary inputs will be focusing on the intended result from the document, which provides the user to set on information extraction mode which can be either simple question, categorization problem, calculation etc. Following are the supporting inputs for the system.

1. Scanned documents (.jpg, .png)
2. Machine-generated documents (word, ppt, excel, pdf)
3. Converted documents (word, pdf)
4. Unstructured documents

4.4 Output

The overall output of the system would be the answer to the relevant query or else document Concept-Relation immediate output if required. This Document Concept-

Relation can be used to import for any supporting system which performs any information extraction task. Altogether following are the outputs of the system.

1. Query answer (Structured sentence, table, graph, numerical answer)
2. Document Concept-Relation (. dls format)
3. Document category
4. Content Summary

4.5 Process

There are main modules where the research implementation is taken placed. Which are namely,

1. Information Extraction Module
2. Query Extractor Module

Primary inputs were given to the Information Extraction Module and it will output the Document Concept-Relation in. DLS format. This module will perform several subprocesses to output DLS such as in the following,

1. Layout Analysis
2. Generate Knowledge-Map
3. Extract Entities
4. Generate Concept-Relation

The output generated through the Information Extraction Module and the second input is given to the Query Extractor Module, which will output the query answer as the output. It involves several sub-processes such as,

1. Encode Query Question
2. Extract Query Answer
3. Presentation

Based on the information extraction mode, the presentation could be varied i.e. category label, summary, calculated answer etc.

4.6 Potential Users of the System

It cannot simply limit the intended user groups for the system, rather this would provide service for a vast range from academic personals to non-technical office employees. Users are not required to have the necessary knowledge in machine learning or conceptualization since the implementation is abstracted from the user. This system will be most useful in digitizing paper-based processes and even processing different groups of machines generated documents. Libraries and Academia users can be the most beneficial user groups from this research.

4.7 Features

Overall system features can be listed as follows, considering the research milestones achieved.

1. Define any document as a concept relation map in. LDS format
2. The system can process any linguistic, any domain document in any subject
3. Depends on less training data
4. User can pick information extraction mode and output will be provided accordingly
5. Based on users' feedback perform transfer learning to adjust to a given subjective environment

4.8 Summary

The limitation and research gaps that have been identified through the literature survey has been addressed by providing solutions as features within the system in scope. As most of the research models require excessive data and sometimes the model is fitting into the trained domain it usually is lacking the generic behaviour. Whereas GIEF for Document Processing has achieved on such limitations.

5. DESIGN OF GIEF FOR DOCUMENT PROCESSING

5.1 Introduction

The design of GIEF for Document Processing is extracted by considering the approach and the processes defined. Also based on the technology and the relevant functionalities it has derives the sub-components and process flows. This chapter describes the overall design of the system using a top-level design diagram. Then it discusses design methodologies in system component-wise.

5.2 Top Level Design

To bring out concept relations, it is required to extract and embed document information somehow. As inspired by transformers, BERT [17], ELMO [18] models, here we are providing a novel framework combining textual, positional and relational information to bring out a pre-trained model, which again will be finetuned for specifically in question answering task.

The given dataset will pass through a preprocessing stage where tokenization occurs, which will be further discussed during the implementation details. According to the framework design, it consists of two main components such that,

1. Information Extraction Module
2. Query Extractor Module

The information extraction module takes in the primary input as any raw document format which can be either in scanned photos or digitized version and then perform the preprocessing. Then it has to go through several subprocesses to provide the document Concept-Relation as the intermediate output. This output will be given to the next component of Query Extractor Module which acts according to the relevant feature extraction mode set and provide the relevant output. Figure 5.1 explains the overview of the top-level design of the GIEF for Document Processing.

The sub-modules inside the Information Extraction Module would be named, Layout Analyzer, Knowledge-Map Generator, Entity Extractor and Concept-Relation

Generator. The first three modules will be processed parallel and the outputs will be taken into Concept-Relation Generator for the processing.

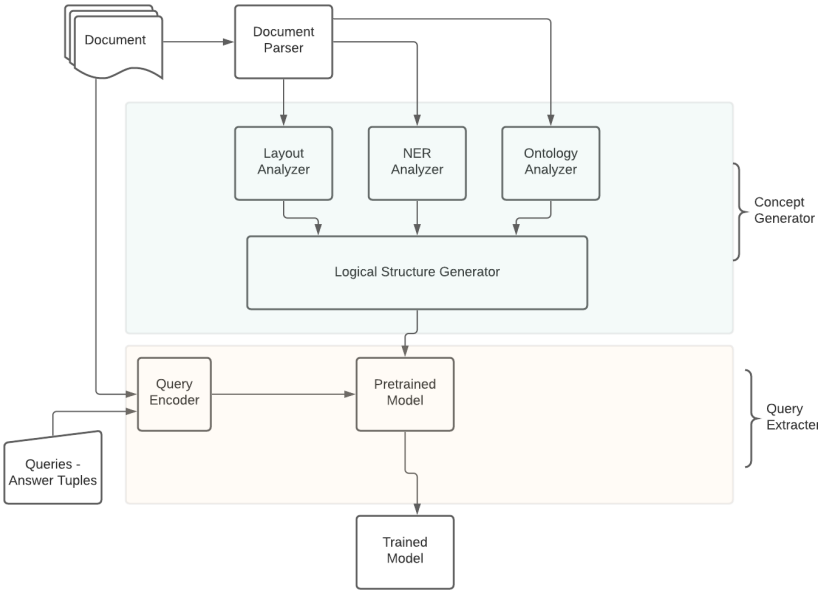


Figure 5.1: Top-level Design of GIEF for Document Processing

The Logical Structure Generator will take the output from document parser and will provide a logical structure which combines textual, positional and entity information with a pre-trained model. Then this will be fine-tuned for the task of question answering.

5.3 Framework Components

5.3.1 Concept-Relation Generator

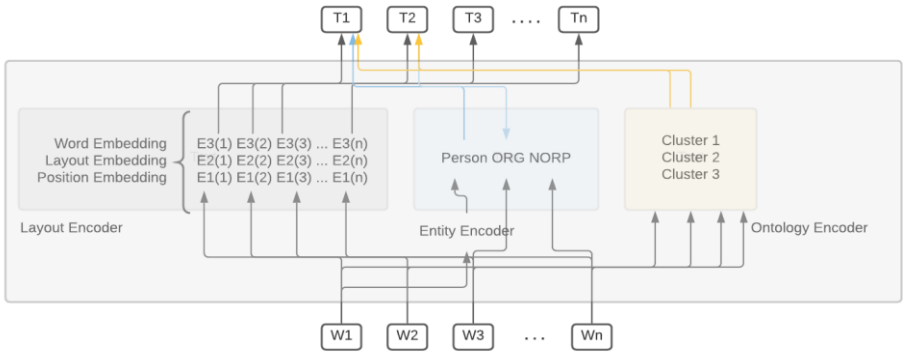


Figure 5.2: Concept Relation Generator

Main feature extraction happens within this module. Input for this module will be served with combined parsed information resulting from the document parser. Collectively, it consists of 3 main sub-modules namely,

1. Layout Analyzer
2. Knowledge-Map Generator
3. Entity Extractor

Each module follows microBERT pattern and results from each will, later on, merge to produce the pretrained model using the poly-encoder mechanism.

This overall architecture of the concept relation generator is inspired by BERT [17] and ELMO [18] studies while it's none other than combining those two. In BERT [17] we will consider bi-directional encoders to produce tokens for each word while following through several embedding layers. It will look at the input document linearly and also will adhere to word-to-word relationships. In ELMO [18] it is defining left-to-right and right-to-left contextual models' encoders and produce tokens from those.

In this study, the document has been looked at different perspectives providing different concepts such as layout, named entity recognition and Knowledge-Map where these three blocks act like left-to-right/right-to-left context modules in ELMO [18]. If we are to look at the internal definition within each submodule, it mimics BERT behaviour in having relevant embedding layers. The novelty in this research lies within how to optimally combine these state-of-art definitions to facilitate the problem statement in hand of providing a framework in document processing.

5.3.1.1 Layout Analyzer

After preprocessing the document, the segmentation descriptors are provided to this module to extract structure related details. The provided document can even be unstructured one hence the segmentation process should occur as accurate as possible to provide correct structure details. This module should identify tables, figures, the hierarchical structure of the document alongside the position of sections, subsections and titles.

As per the output, it will be a form of a map which shows all the structure element relations and the hierarchy in one place. Also, each segment is uniquely labelled so that those can be referred easy when performing information extraction.

Figure 5.2 summarize the input-output relation of this sub-module.

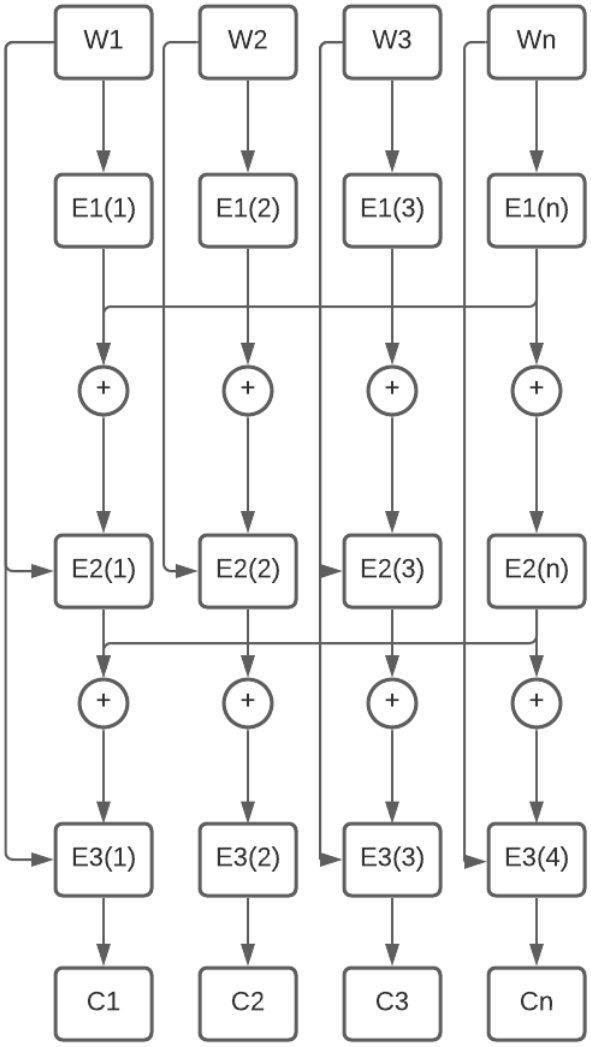


Figure 5.3: Top-level views of Layout Analyzer

Since only considering textual information is not enough, it is better to embed hierarchical and positional information within the word embeddings. Following the base of LayoutLM [30], here we are proposing an additional embedding mechanism

to provide microBERT layout model. Altogether different levels of embedding have been used,

1. Word embedding
2. Positional embedding
3. Layout embedding
4. Font style embedding
5. Segment embedding

It is required to include font-style embedding since some noun phrases might be linked together with same given styles, and also image captions like text are following own styles which contain different contextual information. When it comes to segment embedding, this will consider details such as document headings, document footer, table header, tabular structures etc. For the information extraction purpose, to get more accurate information, we need to consider this information collectively.

Positional embedding contains relations with other words since one word appearing in a given line will have a more impact on a word appearing below perhaps, for example in form layouts. Also, layout embedding will contain positional/layout distribution of words which consider hierarchical structure.

This layout embedding model will generate the layout concept as a vector within the relational schema, which explains the strength of each concept. Upon performing an information extraction task, these concepts will be mapped against the query to extract the relevant answer.

Just like any other encoder mechanism, starting with word tokenization, it will concatenate all the embedding into a dense layer.

5.4.1.2 Knowledge-Map Generator

Knowledge-Map is the best way to represent contextual information hierarchically. Based on the different perspectives we can define different ontologies. Starting from the high-level definition on the document with the topic, titles, sub-titles and respective regions we can derive level-1 Knowledge-Map where each word is marked upon with

belongingness to each entity. Then we can extract Knowledge-Map from structures, which are basically in the binary form of key-value pairs. Those are defined in the level-2 Knowledge-Map. Within the level-3 Knowledge-Map embedding, it extracts complex tabular structural data and encodes words around that.

Combining these 3 types of embedding will provide a generic conceptual model with the sense of Knowledge-Map definition.

If we are to consider the top-view of the sub-module, Knowledge-Map Generator, it takes in words and sentences as inputs and provides the Knowledge-Map of the document as in following Figure 5.3

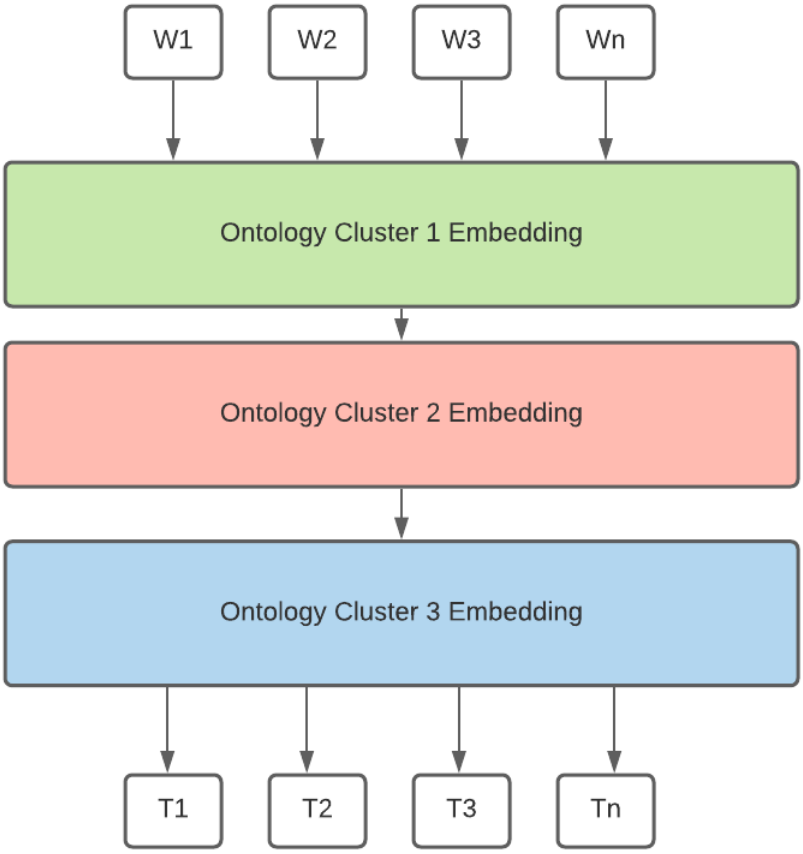


Figure 5.4: Top-level views of Knowledge-Map Generator

Generating a Knowledge-Map is important because when it comes to answering the query, it needs to have some sort of understanding regarding how meaning is

formulated. Semantics in the document content also forms a hierarchical structure based on content concept understanding.

5.4.1.3 Entity Extractor

In the Entity Extractor, the microBERT Named Entity model generates the entity concepts out of the given document. For a considered document, there can be ambiguities in the contextual meanings for the word/phrase. To mitigate that, a multi-level NER embedding has been used.

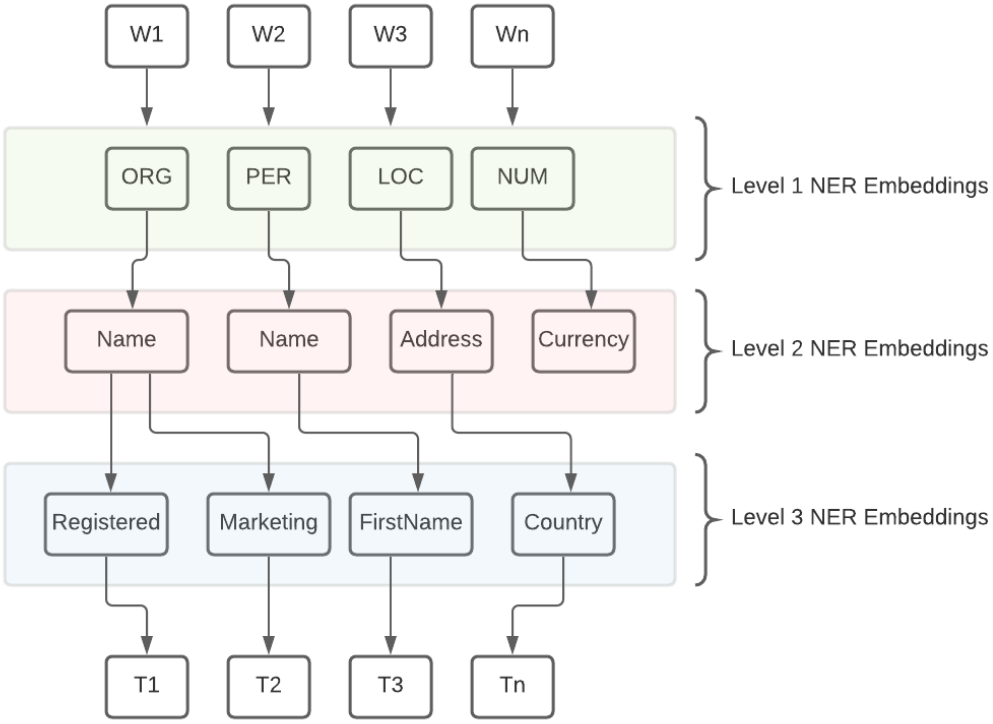


Figure 5.5: Top-level views of Entity Extractor

In the level 1 NER model, it encodes the given sequence of words with high level named entity parse such as ORG, PER, NUM. This will produce level 1 NEW embedding sequence. Examples such as numbers, organization names will require furthermore description of the tagged named entity. To facilitate this, level 2 NER embedding is being used. For example, if an organization name spans several words or number can be span several words. Level 2 specify on NER distribution span out. Not only that, although named entity tag appears as some generic tags such as PER,

ORG, NUM, it is much useful to know the entity type as well. This is considered within level 3 NER embeddings layer. For example, of a given named entity, it can be either an integer or floating type denoting the currency or even in the number of days.

These 3 types of NER embeddings are combined and normalized in microBERT NER to provide entity concept-relation map of the document. In the process of validating the query answer or locating the answer position, this model will be used.

5.4.2 Pre-Trained Model

The GIEF model will be able to provide the logical structure output for any given document. This model can be pretrained for any information extraction task, but with this research, it will specifically focus on question answering task.

5.4.3 Query Extractor

The purpose of the query extractor model is to showcase how the pretrained GIEF model can be fine-tuned for the question answering task, which is the most primitive way of information extraction.

DocVQA dataset has been used for this purpose. This dataset is preprocessed beforehand to generate question-answer pair positions alongside with the other embedding information. For any given document it has been made with question-answer pair and will embed with regional information. The pretrained model is the base for this design, having provided such question-answer pair while masking 10%. Then the model has trained in a semi-supervised manner. This will be tested against different datasets and will be evaluated under different metrics within the Evaluation chapter.

5.5 Summary

This chapter explains on design details on the solution provided in terms of starting from a high-level definition of the design and progressively stepping into submodule design details as well. Considering the required layout, entity information and knowledge modelling information required in document processing these sub-modules have been derived to provide a model for deriving document logical structure definition. Next chapter will discuss the implementation points in detail.

6. IMPLEMENTATION OF GIEF FOR DOCUMENT PROCESSING

6.1 Introduction

This section discusses approaches taken for implementing the design as described in the previous chapter, including output from code of constructing each module. Each of the component-wise details has been discussed and finally provides a notion of how each sub-module being combined to construct the framework. Some modules are using multiple datasets within different phases of training. And also, some implementation code snippets have taken directly from the official documentation git repositories where references have been added whenever necessary. As per the development environment, Kaggle kernels is used with GPU environment.

6.2.1 Layout Analyzer

One of the aims within layout analyzer is to provide a mechanism to express document layout structure along with word embeddings, which again is the key definition of concept relations of GIEF.

This module is consisting of 3 different embeddings such as word embeddings, layout embeddings and positional embeddings. Using LayoutLM [30] and BERT [17], these embeddings has been encoded. Input for this module is the word document in the form of PDF which will finally output a model definition of document layout description.

Before the model generation, preprocessing is performed for each dataset used. Then the embedding layers have been constructed to feed into BERT pretrained model to generate the layout analyzer model.

6.2.1.1 Datasets

To perform the training Receipts, Kaggle dataset [31] has been used. This consist of scanned pdf copies of receipts. It is licensed under CC0: Public Domain. It contains personal receipts collected by the dataset owner within different locations and in different years. All the receipts added are in the form of scanned PDF documents. They have been converted to PDF through a document scanner. Dataset metrics and folder structure is shown in Figure 6.1

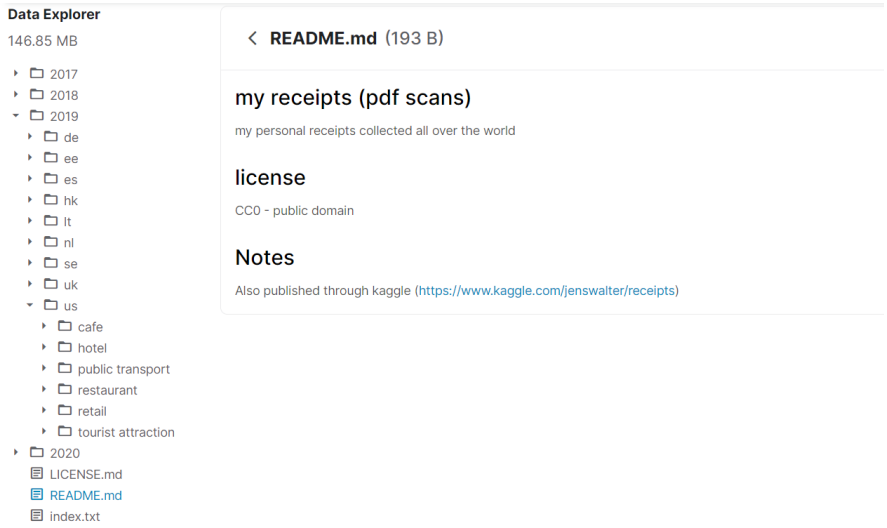


Figure 6.1: Receipts Dataset Structure

Also, the given dataset statistics indicate around 1416 downloads and 0.14 download per view ratio as mentioned on the dataset page on Kaggle.

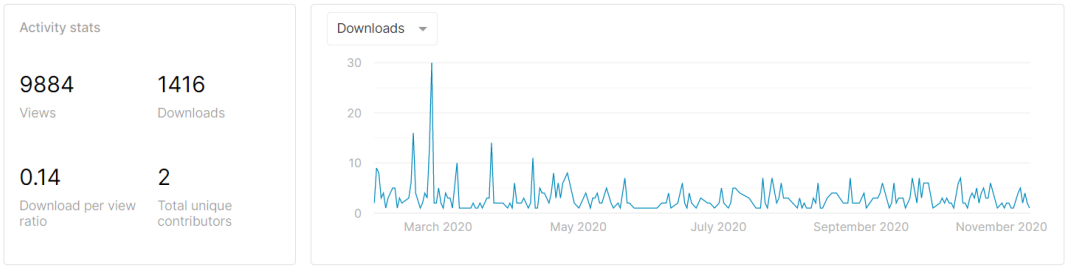


Figure 6.2: Receipts dataset statistics

Reason to use this dataset is to get labelled data for generating OCR tokens to get data with coordinates. For the model training and inference purpose, FUNSD dataset [32] is used.

FUNSD (Form Understanding in Noisy Scanned Documents) dataset [32] is used for miscellaneous purpose including OCR, text detection, form understanding and text detection. It consists of 199 fully annotated forms, 31485 words, 9707 semantic entities and 5304 relations. Following Figure 6.3 illustrates sample word grouping and semantic entity labelling as mentioned on [33].

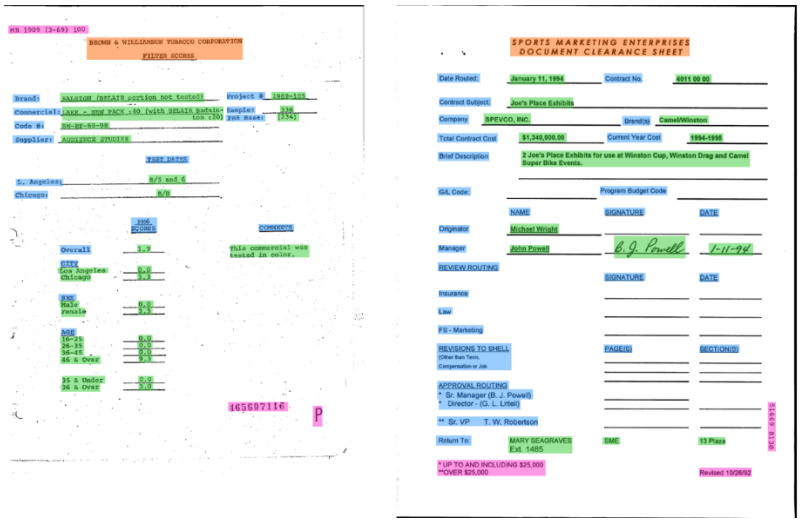


Figure 6.3: Word grouping and semantic entity labelling *[33]*

Annotations are represented in a JSON file, in the example of Figure 6.4 image text information is come in the format shown in Figure 6.5 JSON annotations.

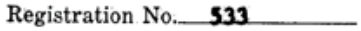


Figure 6. 4 Example image from FUNSD dataset


```
{
    "form": [
    {
        "id": 0,
        "text": "Registration No.",
        "box": [94,169,191,186],
        "linking": [
            [0,1]
        ],
        "label": "question",
        "words": [
            {
                "text": "Registration",
                "box": [94,169,168,186]
            },
            {
                "text": "No.",
                "box": [170,169,191,183]
            }
        ]
    },
    {
        "id": 1,
        "text": "533",
        "box": [209,169,236,182],
        "label": "answer",
        "words": [
            {
                "box": [209,169,236,182
                ],
                "text": "533"
            }
        ],
        "linking": [
            [0,1]
        ]
    }
]
}
```


Figure 6.5: Annotations in JSON for FUNSD example image

6.2.1.2 OCR Preprocessing

First of all, receipt images are required to be preprocessed to prepare for the training. Pytesseract library has been used to generate coordinates and tokens using pdf2image for converting pdf into image form.

Following is an example of a pdf image obtained from the dataset.

THE CHEESECAKE FACTORY
LAS VEGAS

O727a TABLE 82 #Party 1
BRANDON L SvrCk: 32 19:34 12/06/19
DINING ROOM
Separate checks: 2-of-4

1 Great Basin Dr 7.50
1 Louisiana Chicken Pasta 18.50
1 Blue Moon Draft 7.50
1 Original Cheesecake 7.95

Sub Total: 41.45
Tax: 3.42
12/06 20:57 TOTAL: 44.87

Gratuity Not Included
Suggested Gratuity:
22% 9.87
20% 8.97
18% 8.08
15% 6.73

We'd love to hear about your visit!
www.ccfsurvey.com
Enter this code within 5 days:
1092-60171-02027

Join us for Brunch, Sat/Sun 10-2

For to-go orders, please visit
order.thecheesecakefactory.com

Figure 6.6: cheesecake-20191221_003.pdf from the dataset

Then each image will be loaded to Dataframe which yield the following information from each image.

	level	page_num	block_num	par_num	line_num	word_num	left	top	width	height	conf	text	left_scaled	width_scaled	top_scaled	height_scaled	right
4	5	1	1	1	1	1	153	156	39	22	96	THE	248	63	120	17	
5	5	1	1	1	1	2	207	154	131	24	95	CHEESECAKE	335	212	119	19	
6	5	1	1	1	1	3	355	154	90	23	95	FACTORY	575	146	119	18	
8	5	1	1	1	2	1	248	187	37	24	93	LAS	402	60	144	19	
9	5	1	1	1	2	2	300	186	65	24	90	VEGAS	486	105	144	19	
13	5	1	2	1	1	1	33	254	129	23	6	O7F2Z27a	53	209	196	18	
14	5	1	2	1	1	2	194	253	64	24	92	TABLE	314	104	195	19	
15	5	1	2	1	1	3	287	252	24	23	81	62	465	39	195	18	
16	5	1	2	1	1	4	341	252	77	27	75	#Party	553	125	195	21	
17	5	1	2	1	1	5	439	252	10	23	94	1	712	16	195	18	
19	5	1	2	1	2	1	33	287	92	23	96	BRANDON	53	149	222	18	
20	5	1	2	1	2	2	141	288	10	21	33	t=	229	16	222	16	
21	5	1	2	1	2	3	194	285	74	24	53	SvrCk;	314	120	220	19	
22	5	1	2	1	2	4	287	285	24	22	94	32	465	39	220	17	
23	5	1	2	1	2	5	330	285	62	23	94	19:34	535	100	220	18	
24	5	1	2	1	2	6	409	284	103	26	95	12/06/19	663	167	219	20	
26	5	1	2	1	3	1	33	318	156	25	90	DINING	53	253	246	19	
27	5	1	2	1	3	2	220	317	103	24	95	ROOM	357	167	245	19	
30	5	1	2	2	1	1	74	352	104	26	96	Separate	120	169	272	20	
31	5	1	2	2	1	2	194	350	87	24	93	checks:	314	141	270	19	

Figure 6.7: Preprocessed Information from receipt

6.2.1.3 Model Generation

The official LayoutLM [30] repository code is used for the pretrained model, and preprocessing instruction has been followed as mentioned. Results from the preprocessing step on dataset have generated the following data files,

1. train.txt
2. train_box.txt
3. train_image.txt
4. labels.txt

Each of the marked tokens in the document will be categorized among the regions of,

1. header
2. question
3. answer
4. 'O' – something else

layoutlm-base-uncased pretrained model has been used to generate the layout analyzer model. Per GPU train batch size and GPU evaluation batch size has been set to 16 and labels.txt is provided as training examples. By enabling DataParallel, it has utilized all GPUs automatically. Following is a sample from epochs being trained.

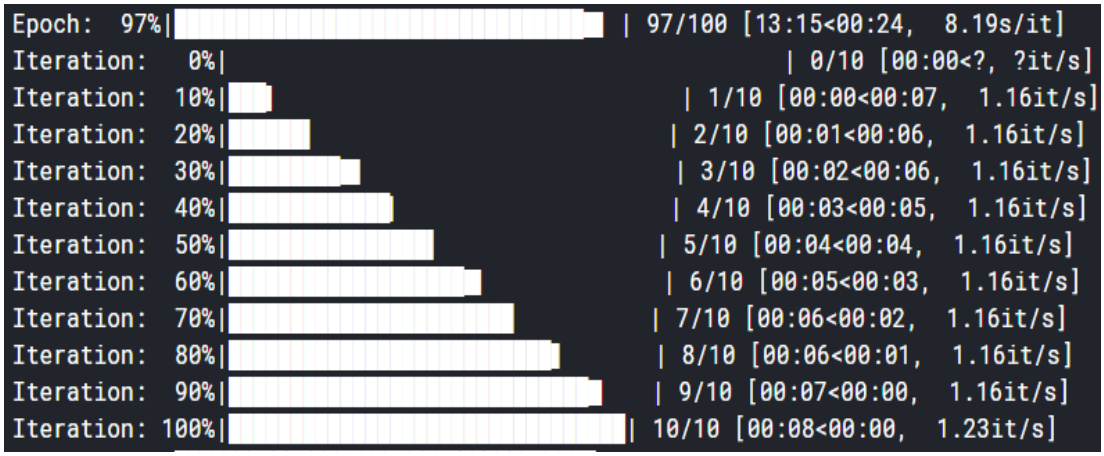


Figure 6.8: Training log of layout analyzer

6.2.2 Knowledge-Map Generator

The knowledge graph is useful in representing information on the document in a more visualized manner with the different nodes connected as concepts. Within this module first knowledge graph is generated, then embedding is performed based on entity definitions in the graph, which is provided into the microBERT model to generate a knowledge-map model to represent conceptual data.

6.2.2.1 Dataset

For the training of knowledge graph model, Stanford Natural Language Inference Corpus (SNLI) [34]. This dataset is consisting of 570152 total pairs and support for many applications including natural language inference evaluation tasks. The dataset activity usage on Kaggle is as follows.

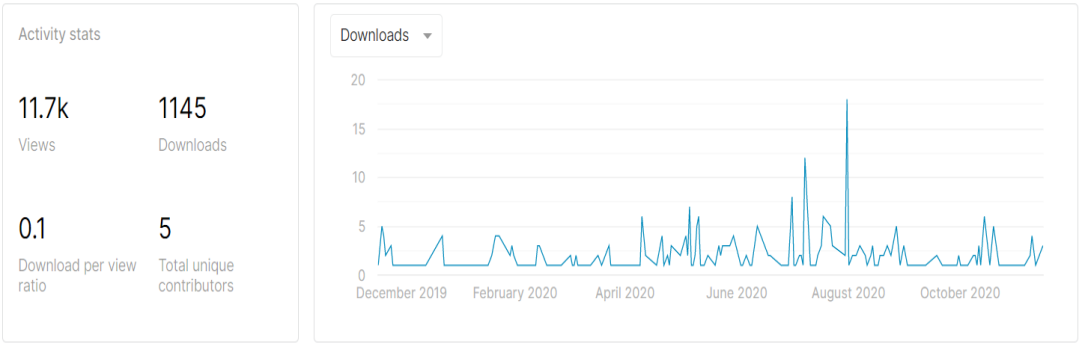


Figure 6.9: SNLI Dataset usage statistics

The given dataset presents the information with the notion of following fields,

1. Sentence 1
2. Sentence2
3. Sentence {1,2} _parse
4. Sentence{1,2}_binary_parse
5. Annotator_labels
6. Gold_label
7. captionID
8. pairID

Following illustrates some of the above column statistic information.

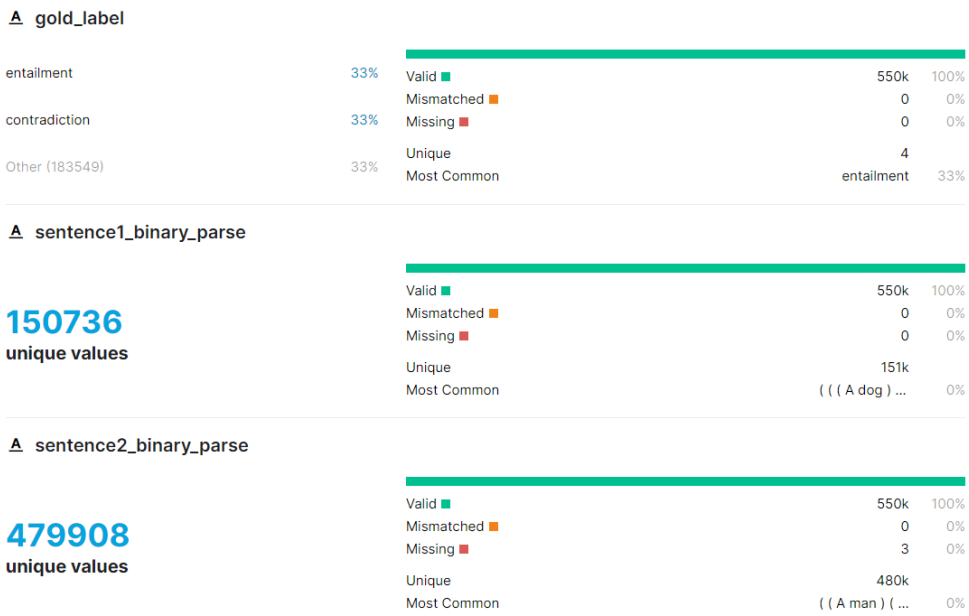


Figure 6.10: SNLI column statistic

```
candidate_sentences['text'].sample(15)
```

```
659                                                                                                  Yo I got bars and I'm not e
ven a rapper
7019          Find out how your fund was used for Typhoon Haiyan in the Philippines. See @DevPeace Haiyan Relief Funds Report http://t.c
o/JwxrX1LsqO
1895                                                 WRAPUP 2-U.S. cable TV companies' shares crushed after Disney disappoints http://t.c
o/wWFACu6NFt
4575          @nalathekoala As a health care professional that deals all gun violence sequalae I consider suicides injuries accidents a
nd homicides
5404                                                                 New post: 'People are finally panicking about cable TV' http://t.c
o/df9FjonVeP
4019    Who is bringing the tornadoes and floods. Who is bringing the climate change. God is after America He is plaguing her\n \n#FARR
AKHAN #QUOTE
4038                                               How is it one careless match can start a forest fire but it takes a whole box to star
t a campfire
5427                               America said it had a war on terrorism. Then police took my AR-15 when I was ready to start shooting
the enemies!
3831                                    @IAFFLocal4416 no lives lost except those of first responders or the public they're charged with
protecting?
```

Figure 6.11: Dataset Sample

6.2.2.2 Preprocessing

Dataset is loaded and then it has been segmented into sentences. The result is shortlisted to have subject-object pairs to construct knowledge graph node relationships. For example, the sentence “0.04% of the votes were invalid” will be tokenized as in Figure 6.12.

```
0.04 ... nummod
% ... nsubj
of ... prep
the ... det
votes ... pobj
were ... ROOT
invalid ... acomp
. ... punct
```

Figure 6.12: Segmentation of the example sentence

6.2.2.3 Model Generation

To build the knowledge graph, some prior information needs to be extracted, for example, entity extractions. By using POS tagging nouns and proper nouns have been extracted as entities for the knowledge graph. Upon performing entity extraction on the example sentence of “0.04% of the votes were invalid” it will result in the entities as “0.04 %” and “votes”.

```
get_entities("0.04% of the votes were invalid")
```

```
['0.04  %', 'votes']
```

Figure 6.13: Entity Extraction result for the example

Likewise, all the entity pairs have been extracted from the dataset. During the processing these entity-pairs has been extracted, which are in the form of subject-object pairs for the nodes. Some of the entity pairs are illustrated in Figure 6.14.

```
[['Three  people', 'heat wave'],
 ['I', 'SOUTH TAMPA'],
 ['19 Florida I', 'count'],
 ['Bago Myanmar We', 'Bago Myanmar #'],
 ['Damage', 'multi car crash #'],
 ['up  man', ''],
 ['I', 'fruits'],
 ['Summer', ''],
 ['car', ''],
 ['', '']]
```

Figure 6.14: Extracted entity pairs sample

To connect the nodes in the knowledge graph, verbs are used since the verb is considered as the linking point between subject-object pairs. Using spaCy rule-based matching such relations were defined. Upon providing the same example “0.04% of the votes were invalid” to get the relation it returns “were invalid” as the related term.

```
get_relation("0.04% of the votes were invalid")
```

```
'were invalid'
```

Figure 6.15: relation result for the example sentence

After extracting all the relations from the dataset, Figure 6.16 shows the frequency of the terms collected.

```
pd.Series(relations).value_counts()[:50]
```

```
is              163
's               78
have             48
be               43
'm               38
are              38
was              37
said             28
|                27
liked            25
_                25
In               25
been             25
know             24
Latest           23
want             23
```

Figure 6.16: Relations frequency

Then the entities and relation predicates have been loaded to data frames, which will then use with network library for creating the network in a form of the directed graph. The plot of the network is illustrated in Figure 6.17.

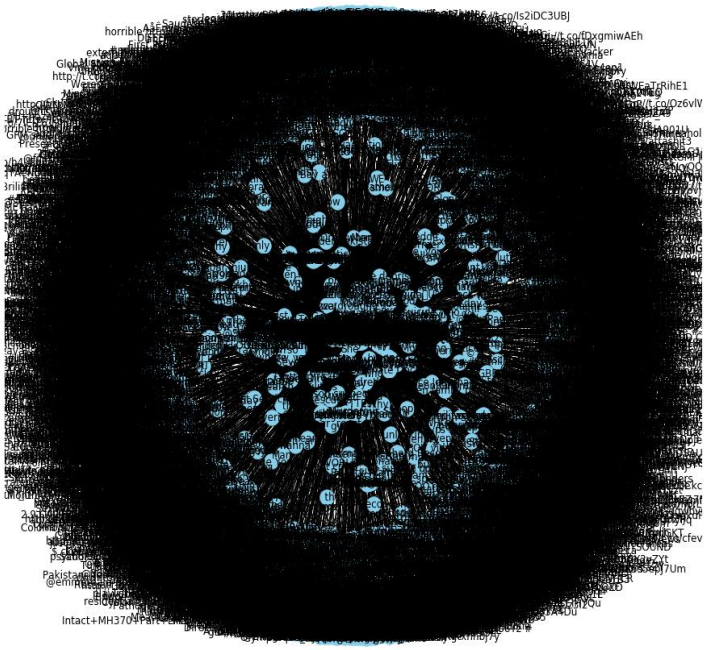


Figure 6.17: Knowledge Graph Representation

6.2.3 Entity Extractor

Document text can represent different levels of named entity mapping. To represent such conceptual information, Entity Extractor is being used. Different levels of entity embeddings have been fed into microBERT model to train on encoding named entity information from the given document.

6.2.3.1 Datasets

To train the Entity Encoder, Kaggle med_train [35] dataset and Kaggle entity-annotated-corpus [36] has been used. Dataset metrics from train.csv for the dataset med_train is illustrated as follows.

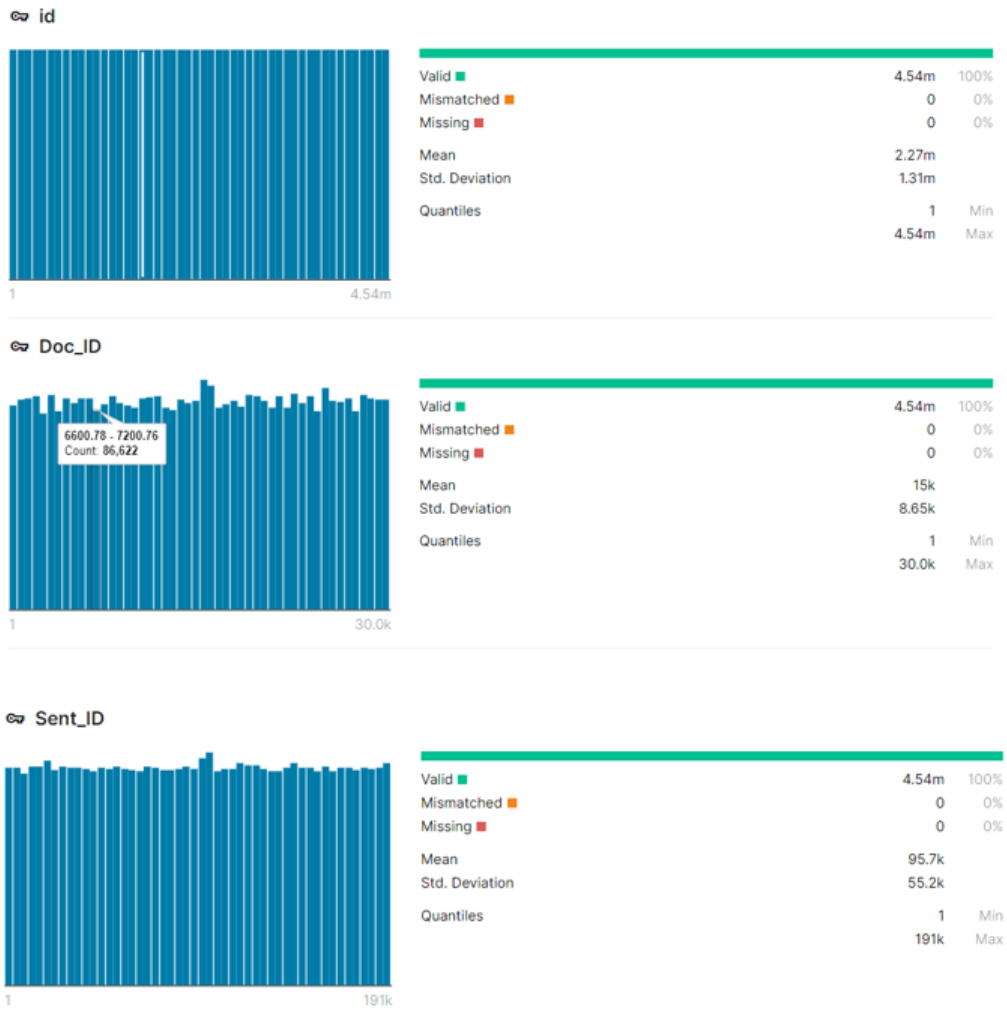


Figure 6.18: med_train dataset statistics

entity-annotated-corpus dataset is surprising around 31k downloads and having 72 total unique contributors to the dataset. Its usage metrics are shown in Figure 6.10.

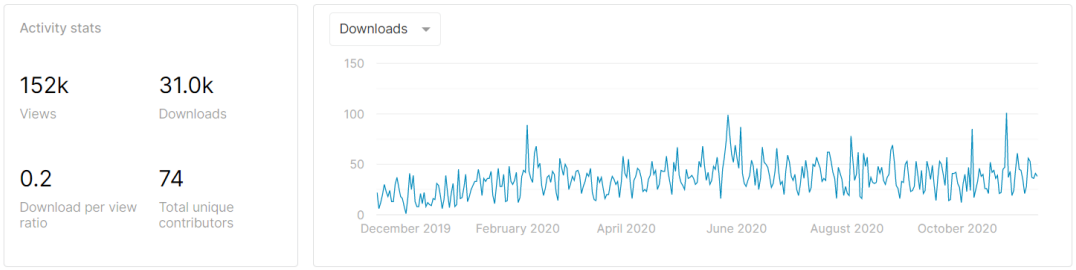


Figure 6.19: entity-annotated-corpus dataset statistics

This dataset comprised of high-level entity tagging information. It is an annotated corpus mainly used for various kernels in named entity recognition applications. Using GMB (Groningen Meaning Bank) corpus it has performed entity classification. Total word count in this dataset is of 1354149 and the target data column is named as “tag”. There is several entity-tag naming provided,

1. ORG – organization
2. GEO – geographical entity
3. NAT – a natural phenomenon
4. TIM – time indicator
5. PER – person
6. EVE – event
7. GPE – geopolitical entity
8. ART – artifact

Example of tagged named entities is illustrated in Figure 6.11 as in below.

```
'O': 1146068', geo-nam': 58388, 'org-nam': 48034, 'per-nam': 23790, 'gpe-nam': 20680, 'tim-dat': 12786, 'tim-dow': 11404,
'per-tit': 9800, 'per-fam': 8152, 'tim-yoc': 5290, 'tim-moy': 4262, 'per-giv': 2413, 'tim-clo': 891, 'art-nam': 866, 'eve-
nam': 602, 'nat-nam': 300, 'tim-nam': 146, 'eve-ord': 107, 'per-ini': 60, 'org-leg': 60, 'per-ord': 38, 'tim-dom': 10, 'per-
mid': 1, 'art-add': 1
```

Figure 6.20: Number of tagged entities

6.2.3.2 Preprocessing

The loaded dataset is illustrated as follows.

	id	Doc_ID	Sent_ID	Word	tag
0	1	1	1	Obesity	O
1	2	1	1	in	O
2	3	1	1	Low-	O
3	4	1	1	and	O
4	5	1	1	Middle-Income	O
5	6	1	1	Countries	O
6	7	1	1	:	O
7	8	1	1	Burden	O
8	9	1	1	,	O
9	10	1	1	Drivers	O
10	11	1	1	,	O
11	12	1	1	and	O
12	13	1	1	Emerging	O
13	14	1	1	Challenges	O
14	15	1	1	.	O
15	16	1	2	We	O
16	17	1	2	have	O
17	18	1	2	reviewed	O
18	19	1	2	the	O
19	20	1	2	distinctive	O
20	21	1	2	features	O

Figure 6.21: NER dataset annotation

First of all, tokens have been generated by identifying sentences, following is a sample of sentence performing word tokenization.

```
['Excellent',
 'reproducibility',
 'of',
 'laser',
 'speckle',
 'contrast',
 'imaging',
 'to',
 'assess',
 'skin',
 'microvascular',
 'reactivity',
 '.']
```

Figure 6.22: Example of a sentence

```
['O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O']
```

Figure 6.23: Labels for the example sentence

6.2.3.3 Model Generation

Several levels of named entity tokens have been embedded for the same document. Then each of the encoded information is used together to generate the entity extractor model.

The training data log is as follows.

```
Epoch:  33%|[redacted]       | 1/3 [25:56<51:52, 1556.08s/it]

Validation F1-Score: 0.7611618960223062

Average train loss: 0.028646908162185944
Validation loss: 0.036002234922255544
Validation Accuracy: 0.43388096432552986
```

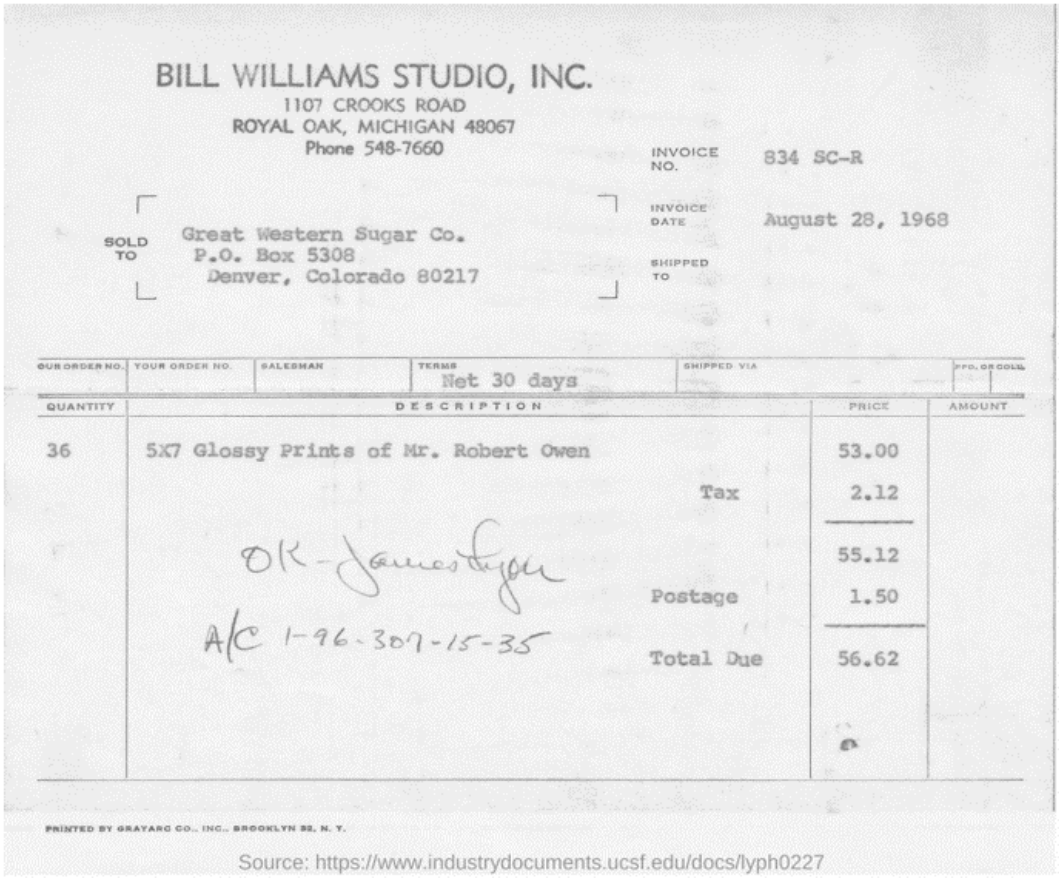
Figure 6.24: Entity Extractor training log

6.2.4 Query Extractor Module

The pretrained model obtained upon merging aforementioned sub-model is fine-tuned within this module for the question-answering task.

6.2.4.1 Datasets

The Document Visual Question Answering (DocVQA) [37] dataset is being used as the primary dataset for the processing of the query extractor module. This dataset mainly facilitates in visual question answering based on document images. It has a richest of layout structure, which provides the uniqueness among other question answering datasets. The dataset was presented on Document Visual Question Answering (DocVQA) [38].



Source: https://www.industrydocuments.ucsf.edu/docs/lyph0227

What is the amount of total due? 56.62

What is the Invoice Number? 834 SC-R

What is the description of the quantity? 5x7 Glossy Prints of Mr- Robert Owen

Figure 6.25: Example Document from DocVQA with Question-Answer

6.2.4.2 Preprocessing

The DocVQA dataset is then preprocessed to be in a question-answer format as illustrated in Figure 6.26.

```
train_data = [
    {
        'context': "This tweet sentiment extraction challenge is great",
        'qas': [
            {
                'id': "00001",
                'question': "positive",
                'answers': [
                    {
                        'text': "is great",
                        'answer_start': 43
                    }
                ]
            }
        ]
    }
    ]
```

Figure 6.26: Question-Answer Format

Sample question-answer set is as in below Figure 6.27

	ArticleTitle	Question	Answer	DifficultyFromQuestioner	DifficultyFromAnswerer	ArticleFile
0	Abraham_Lincoln	Was Abraham Lincoln the sixteenth President of...	yes	easy	easy	S08_set3_a4
1	Abraham_Lincoln	Was Abraham Lincoln the sixteenth President of...	Yes.	easy	easy	S08_set3_a4
2	Abraham_Lincoln	Did Lincoln sign the National Banking Act of 1...	yes	easy	medium	S08_set3_a4
3	Abraham_Lincoln	Did Lincoln sign the National Banking Act of 1...	Yes.	easy	easy	S08_set3_a4
4	Abraham_Lincoln	Did his mother die of pneumonia?	no	easy	medium	S08_set3_a4

Figure 6.27: Dataset Visualization

6.2.4.3 Model Generation

The pretrained model output has been provided to question-answering model with the requirements as mentioned in following. Those are,

1. reprocess_input_data = True
2. overwrite_output_dir = True
3. learning_rate = 5e-5
4. num_train_epochs = 3
5. max_seq_length = 192

6. doc_stride = 64
7. fp16 = False

6.3 Summary

Within this chapter, implementation details have been discussed with considering how each of the submodules has been implemented. Each of the submodules has been processed through dataset selection, preprocessing and model generation, which has been discussed in detail. Within the following chapter, it will derive the evaluation strategy in order to state the validity and accuracy of the given solution by comparing different metrics and models on different datasets.

7. EVALUATION

7.1 Introduction

This section focuses on the experimental design and the discussion on the results obtained throughout the experiment. Experimental design, evaluation strategy, experimental sample data, experimental results and concluding discussion on results has been considered throughout this section respectively. As part of evaluation strategies benchmark datasets such as SQUAD and DocVQA has been addressed with a notion of reasons to use those datasets for performance evaluation. Also among the metrics noted in evaluation strategy, several have been noted such as Precision and Recall, F1 Score, ANLS and Accuracy. In order to compare this solution with existing other solutions, bert-base, bert-large models have been used since those are closer to the provided solution and are better comparison models due to that.

7.2 Experimental Design

To evaluate the theoretical framework and other state-of-art solutions, the experimental design needs to be carried out. This will be helpful to determine which one is the best model, whether the hypothesis is proved up to expectations.

7.3 Evaluation Strategy

With the evaluation strategy, the objectives for the research has been verified using several evaluation matrices and compared among different theoretical frameworks and different approaches. Also, the evaluation performed on different benchmarking datasets to showcase its generic behaviour. The models such as bert-base and bert-large have been used as comparison tools with this solution to provide a better insight into the performance evaluation.

7.3.1 Benchmark Datasets

The solution has been evaluated with 2 types of datasets, SQuAD and DocVQA. It is tested by comparing these two datasets with the known state-of-art model evaluation results registered. Having tested under such benchmarked dataset provides a better insight into the solution accuracy.

7.3.1.1 SQuAD [39]

The Stanford Question Answering Dataset (SQuAD) [39] contains a pair of questions and answers. It mainly targets to facilitate question answering task. Based on the text passage given, it provides answers to some questions and can use to train models on similar tasks.

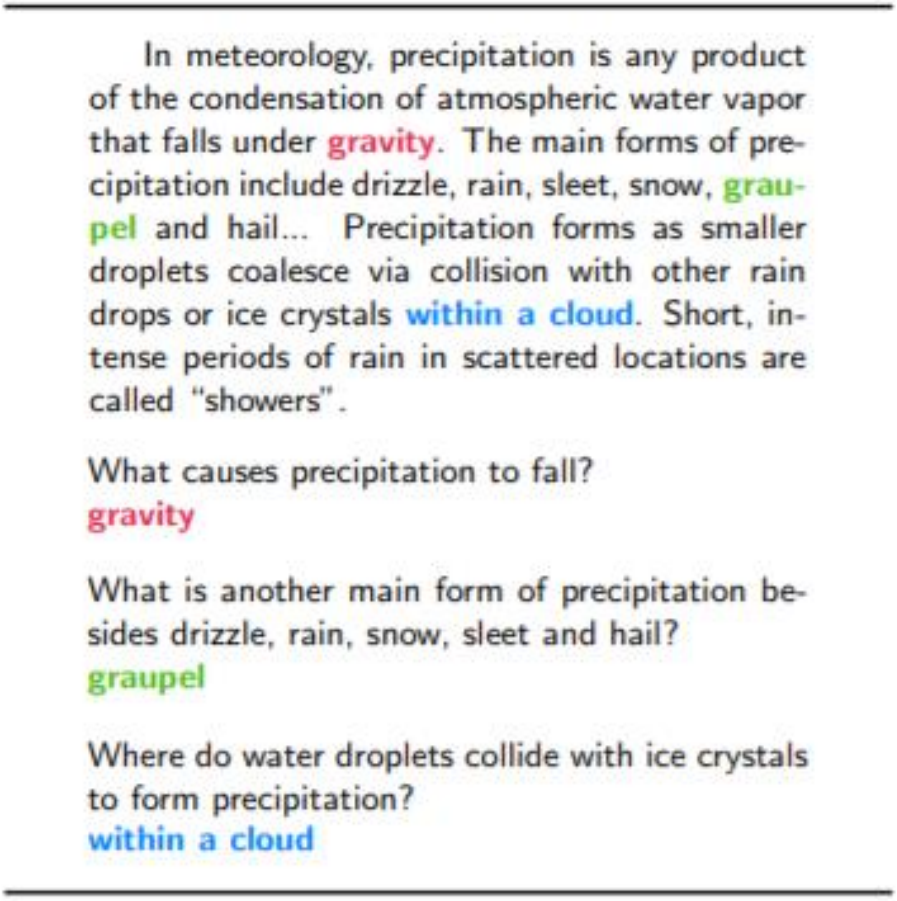


Figure 7.1: Question Answer Pair from SQuAD dataset *[39]*

The dataset has been created from around 10000 Wikipedia articles by sampling 536 and selecting 23,215 of individual paragraphs. For the training set, 80% of the articles have been allocated. Around 10% was separated into development set while the rest of 10% given for the testing set.

7.3.1.2 Why SQuAD?

The SQuAD dataset is well known for evaluation of question-answering task. Among several other datasets existing for the evaluation purposes, SQuAD has some major advantages visible. One is the hugeness of the dataset, as this provides around 100,000+ questions altogether.

Also, when it comes to other document answering datasets, some are focusing on providing the answer by referring to multiple documents. But comparatively the SQuAD dataset provides the dataset in a way such that question-answer pair is provided based on one given paragraph. As this GIEF for document processing mainly focuses on answering queries from a given one document, this is the most suitable dataset to evaluate this solution.

Only extracting answers just as appearing on the document is not enough. Sometimes reasoning is required in intelligent query extractions, whereas we need to use such datasets which require reasoning. As the SQuAD provides answers for complex questions event it fulfils this requirement and better for evaluating such understanding in the model.

7.3.1.3 DocVQA [37]

Document Visual Question Answering (DocVQA) dataset is also selected to evaluate GIEF for document processing to make sure that this solution works well with different types of testing data provided. As the final model, the combined query answering model is trained with DocVQA dataset, the same test dataset is used to evaluate the accuracy. DocVQA dataset is hosted in DocVQA challenge in CVPE2020 workshop on Text and Documents in the Deep Learning Era [40].

This dataset consists of document images which targeted for the task of visual question answering. It contains 12000+ document images and has around 50000 questions based on those images. One document may contain several question-answer pairs defined as in the following Figure 7.2.

The DocVQA dataset provides a better representation of different questions based on the same document, which fits best for question answering model. The diversity of

question inference categories lies up to very complex derivations questions as well which are considering the layout of the document.

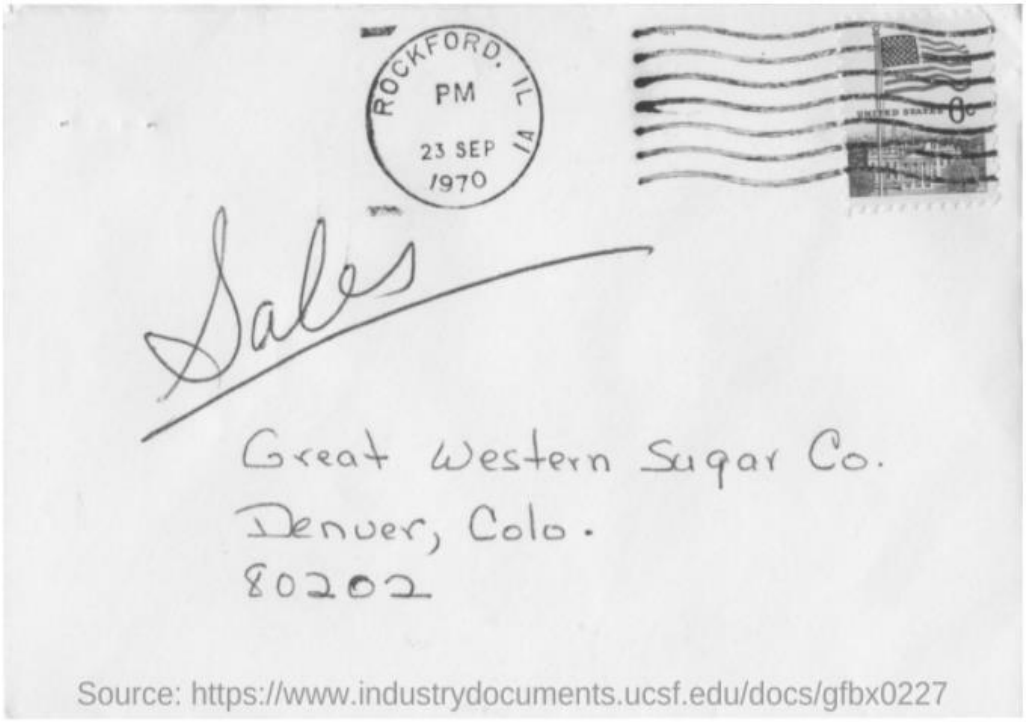


Figure 7.2: Sample Question-Answer pairs from DocVQA dataset *[37]*

7.3.1.4 Why DocVQA?

The purpose of GIEF for Document Processing is to provide a solution which is capable to answer ad-hoc queries for any unseen document, which resolves the need to excessive training on the same domain-related documents. For example, if we are to provide generic behaviour among different domains same time such as financial and

medical both, the traditional solutions will be a focus on training the model on those particularly targeted datasets while the solution will eventually fail on the untrained domain such as political.

Having this objective on hand, it is required to evaluate a more generic dataset which consists of ad-hoc queries for information on given document images and answers examples provided just the same as human users. This dataset covers different reasoning mechanisms which fit best for evaluating the proposed solution.

7.3.2 Benchmark Models

To evaluate the solution performance, several models are compared within such as bert-base, bert-large among different metrics. These are used as the comparison schemes to provide an insight into where the current solution stands in terms of performance.

7.3.2.1 BERT base

The BERT base model consists of 12 transformer layers, 12 attention heads summing up into 110 million parameters. There are several BERT-base models in practice, such as,

1. BERT-Base, Uncased
2. BERT-Base, Cased
3. BERT-Base, Multilingual Cased
4. BERT-Base, Multilingual Uncased
5. BERT-Base, Chines

In Cased versions, the true case of the words and any accent markers are kept unchanged and also taken into consideration. Within uncased versions before the WordPiece tokenization, the text has been converted to the lower case and removed any accent markers. Usually, uncased models are recommended in practice. All of the aforementioned models' shares around the same number of layers, hidden units, attention heads and the number of parameters but different in the optimized pretraining task.

7.3.2.2 BERT large

The BERT large model contains 24 transformer blocks, 16 attention heads resulting in 340 million parameters. BERT-large model again appear in variations such as

1. BERT-Large, Uncased (Original)
2. BERT-Large, Uncased(Whole Word Masking)
3. BERT-Large, Cased (Original)
4. BERT-Large, Cased (Whole Word Masking)

Model	SQUAD 1.1 F1/EM	Multi NLI Accuracy
BERT-Large, Uncased (Original)	91.0/84.3	86.05
BERT-Large, Uncased(Whole Word Masking)	92.8/86.7	87.07
BERT-Large, Cased (Original)	91.5/84.8	86.09
BERT-Large, Cased (Whole Word Masking)	92.9/86.7	86.46

Table 7.1: BERT-Large Models Accuracies *[41]*

7.3.3 Metrics

Several metrics have been used to evaluate the performance of their solution also with comparison to other models. Different metrics have used to evaluate with given models and also even dataset wise different metrics has been used.

		Predicted	
		Negative	**Positive**
Actual	**Negative**	True Negative	False Positive
	Positive	False Negative	True Positive

Figure 7.3: Confusion Matrix

Usually, within any research, it is the goal to minimize false positives and false negatives as much as possible and the derived matrics also share the same goal.

7.3.2.2 Precision and Recall

The metric of Precision and Recall will be used to evaluate the overall solution. Calculation of precision and recall involve in true positive and true negative counts. Figure 7.1 explains the intuition behind this concept. When the Precision-Recall metric applied to the task of information retrieval, precision represents the result relevance measure where recall denotes on actual ground truth values. The precision value doesn't decrease with the recall.

The equations to calculate Precision and Recall is shown in Eq 7.1 - Eq 7.4 respectively. The precision focus on how accurate or precise the model is comparing to the once that has been predicted as positive and taken into consideration of ground truth.

$$\text{Precision} = \frac{\text{True Positive}}{\text{True Positive+False Positive}} \quad \text{(Eq 7.1)}$$

$$\text{Precision} = \frac{\text{True Positive}}{\text{Total Predicted Positive}} \quad \text{(Eq 7.2)}$$

Within the recall calculation, it focuses on how much of actual ground truth ones have been captured among true positive ones.

$$\text{Recall} = \frac{\text{True Positive}}{\text{True Positive+False Negative}} \quad \text{(Eq 7.3)}$$

$$\text{Precision} = \frac{\text{True Positive}}{\text{True Actual Positive}} \quad \text{(Eq 7.4)}$$

In the context of question answering, above formal equations and be derived as in the following Eq 7.5 and Eq 7.6 respectively.

$$\text{Precision} = \frac{1.0 * \text{num\_same}}{\text{len(predicted tokens)}} \quad \text{(Eq 7.5)}$$

$$\text{Recall} = \frac{1.0 * \text{num\_same}}{\text{gold tokens}} \quad \text{(Eq 7.6)}$$

Predicted tokens mean none other than the answer provided by the model and gold tokens are the ground truth values. For example in the document context containing a sentence like "Today is going to be a rainy day and tomorrow it will be snowy" the gold answer or the ground-truth value provided by the dataset is "rainy". Then the

queried answer through the given model can be for example “rainy day”. To calculate precision and recall in line with the formal formulas, True Positive considered as the number of predicted answer tokens appears as the same as ground truth values provided, false-positive are taken from the difference between predicted tokens length and ground truth tokens length. False negatives are calculated from the difference between the length of gold tokens and same tokens numbers. The relevant equations are visualized in below. These precision-recall calculations have considered matching words/tokens in predicted and ground-truth answers and length of each also considered appropriately.

$$\text{True Positive} = \text{num\_same} \quad \text{(Eq 7.7)}$$

$$\text{False Positive} = \text{len}(\text{pred\_tokens}) - \text{num\_same} \quad \text{(Eq 7.8)}$$

$$\text{False Negative} = \text{len}(\text{gold\_tokens}) - \text{num\_same} \quad \text{(Eq 7.9)}$$

7.3.2.3 F1 Score

The metric of F1 will be used to evaluate the overall solution. F1 score is being calculated by combining precision and recall results, which explains by the Eq 7.5. This metric illustrates a more balanced calculation between prevision and recall.

$$\text{F1} = 2 \text{ x } \frac{\text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}} \quad \text{(Eq 7.5)}$$

The F1 score is also known as F-measure or balanced F-score. This can be seen as a sort of weighted average provided among Precision and Recall values. It will appear as a value between 1 and 0 where the best value is considered as 1. In the best score of F1, it indicated perfect precision and recall. This F1 Score is more popular within Natural Language Processing researches, for example, word segmentation and named entity recognition [42].

7.3.2.4 Average Normalized Levenshtein Similarity (ANLS)

The metric of Average Normalized Levenshtein Similarity (ANLS) is used to evaluate the evaluation results on DocVQA dataset. The ANLS add a slight penalty to consider OCR mistakes as well. A threshold of 0.5 is used here to determine whether a wrong answer provided vs correct answer provided. The following Eq 7.6 shows how ANLS

is being calculated. This metric calculation is adapted from Task 3 of DocVQA evaluation [38].

$$\text{ANLS} = \frac{1}{N}\sum_{i=0}^{N}\left(\max_{j} s(a_{ij}, o_{q_i})\right) \quad \text{(Eq 7.6)}$$

$$s(a_{ij}, o_{q_i}) = \begin{cases} \left(1 - \text{NL}\left(a_{ij}, o_{q_i}\right)\right) & \text{if } \text{NL}(a_{ij}, o_{q_i}) < \tau \\ 0 & \text{if } \text{NL}(a_{ij}, o_{q_i}) > \tau \end{cases} \quad \text{(Eq 7.7)}$$

Here the net output and the ground truth answer is compared in Eq 7.6. N is the total number of questions provided where a_{ij} is the ground truth answer with $i = \{0, 1, \ldots N\}$ and $j = \{0, 1, \ldots M\}$, while M is set for the total number of ground-truth answers for each question. The model answer is denoted by o_{q_i}.

7.4 Experimental Results

The solution is being evaluated under the aforementioned benchmark datasets and evaluation metrics and compared with other existing state of art solutions. Experiment results are compared with several BERT based models. The following Table 7.1 shows Accuracy and ANLS performance revaluation results on DocVQA dataset on test and validation sets comparing this solution with bert-base, bert-large models.

Model	**ANLS**		**Accuracy**	
	Validation	**Test**	**Validation**	**Test**
bert-base	0.556	0.574	45.6	47.6
bert-large	0.594	0.610	49.28	51.08
GIEF for Document Processing	0.561	0.583	49.31	52.78

Table 7.2: ANLS and Accuracy Comparison on DocVQA dataset

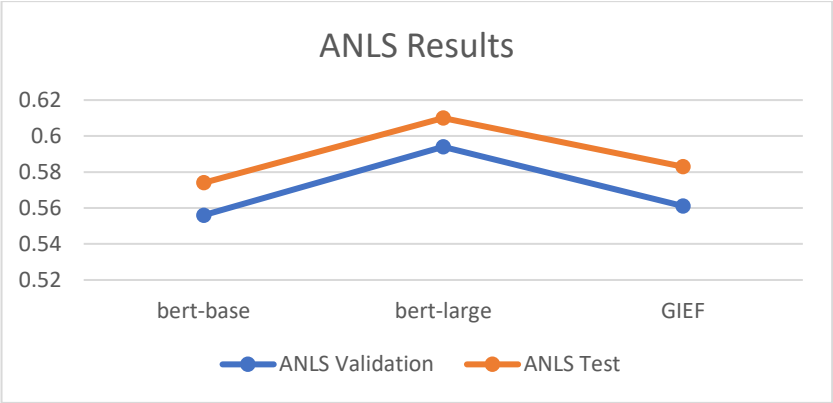


Figure 7.4: ANLS Results on Validation and Test sets

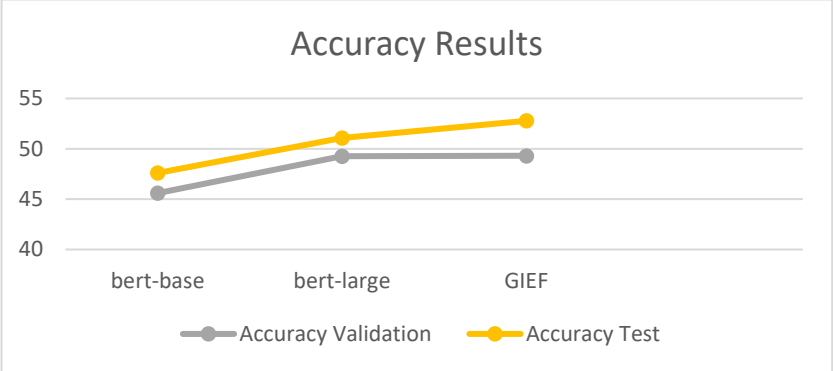


Figure 7.5: Accuracy Results on Validation and Test Sets

The F1 scores comparing different models is illustrated as in follows. The models BERT (single model), SLQA+ (single model), BERT finetune baseline and GIEF for document processing are compared against the F1 scores obtained on SQUAD 2.0 dataset.

Model	**F1 Score**
BERT (single model)	83.01
BERT finetune baseline	86.096
GIEF for Document Processing	87.01

Table 7.3: F1 Scores on SQUAD 2.0 Dataset

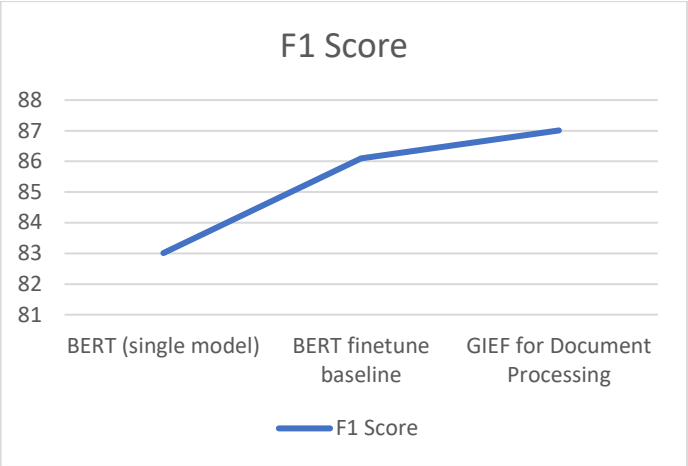


Figure 7.6: F1 Score Graph on SQUAD Dataset

7.5 Discussion on Results

DocVQA and SQUAD datasets have been used for evaluation purpose since those are well-known benchmarks for question-answering evaluation mechanism. Also, metrics such as F1 and ANLS has been derived among different comparison models such as bert-base and bert-large. The main reason to compare with bert model varieties without considering other popular question answering models is that since this solution adapted the baseline architecture from BERT it is better to compare with the parent architecture rather than comparing with solutions provided by different model architectures.

As per experiment results on Table 7.2 it can be seen that the overall accuracy and ANLS metric on DocVQA dataset are comparatively slightly higher than those of bert-base model, but it is less than the same on bert-large models. Since the bert-large dataset is being trained and fine-tined on a larger dataset and being more optimized it is possible to have a lesser performance than it. Since this solution is trained by considering the layout, entity and knowledge modelling varieties that occur within a document, it shows a higher accuracy metrics than the bert-base models.

When considering F1 scores evaluated on SQUAD dataset among with BERT model and BERT finetune baseline models, GIEF for Document Processing showcases a higher performance comparatively. This showcase the generic behaviour on the provided solution concerning how well it can answer queries on documents on different complexities.

7.6 Summary

In overall, this chapter expresses an evaluation strategy performed on this research with accuracy metrics numbers and comparisons with other models. With this, it can be clearly visualized that this solution stands on better grounds in providing a more accurate and efficient solution for the problem statement on deriving a logical structure of concept relations from a given document. Next chapter will add concluding remarks to this study with a notion of limitations and future works stated.

8. CONCLUSION AND FUTURE WORK

8.1 Introduction

This section follows concluding remarks followed by limitations and future work in this research. Short summarization over research process is being discussed providing insights on key decisions made in milestones. The overall evaluation results will be also discussed in summary with a notion of key points for success and how far objectives have been achieved. Followed by the summary this chapter will finalize the thesis content.

8.2 Concluding Remarks

The objective of this study is to provide a suitable solution in deriving logical structure from any given document, which is capable in specifically query answering task. Keeping this objective in mind, first of all, background for this particular problem has been explored with a thorough study of existing literature. By analyzing unresolved problems and future works mentioned in such research the hypothesis of the research has been set to come up with a generic framework for document processing.

The key idea of this research is to provide a mechanism to derive concept relations in a form of the logical structure of the given document. The existing and usual document information extraction mechanisms involved with performing OCR on document text and then while performing natural language processing tasks then perform any desired tasks, whereas the model needs to be specifically trained upon such tasks. But rather than looking at a given document in the perspective of just mere text, we should consider layout, entity and knowledge map concept as well. Altogether these concepts will be in relation to each other, upon carefully deriving such concept-relations, a particular model can be fine-tuned to various natural language information extraction tasks. As per the scope of this study, only the question-answering task is considered in showcasing the model behaviour in practice.

In order to follow up on the implementation details, several technologies have been analyzed such as transformer models and BERT based models. Upon providing design and implementation details, the overall module has been divided into several dub

modules addressing layout extraction, entity tagging and knowledge map generation from documents. Combining these tasks provided a more generic model describing the logical structure of the document which later on fine-tuned on question answering task. As per inspirations, BERT and ELMo model architectures has been adapted but in a novel way, since no such study currently exists in combining these two, instead always comparisons made between the two states that the BERT provide a better language model comparatively. The novelty of this research on providing an adapted architecture combining these two state-of-art language modelling techniques.

Upon diving into experimental design, starting from the dataset evaluation several experiments have been carried out. Datasets such as SQUAD 2.0 and DocVQA has been used to evaluate the overall performance over metrics such as F1 score, accuracy and ANLS providing scores 87.01,52.78 and 0.583 respectively. The F1 score, which is 87.01 showcase that the provided solution achieves the expected objectives in deriving a generic model fitting for any question-answering task based on documents.

8.3 Limitations and Future Works

Currently, this solution works well in English language only. Although providing multi-linguistic capabilities was also one of the tasks within this research, this has been reserved for the future works as a more focus provided on providing a generic solution concerning different layout and textual entity variations.

Since this model has been focused on question answering tasks only within this research, but definitely can be fine-tuned for some other NLP tasks such as text summarization and document classifications since a generic logical structure can be extracted.

8.4 Summary

This section consists of concluding remark on this solution presentation of the thesis, having discussed on overall research milestones together with limitations and future work to note. Additional resources which excluded from the chapter contents will be followed after the References section as several Appendices.

REFERENCES

[1] A. Ritter, S. Clark, Mausam and O. Etzioni, "Named Entity Recognition in Tweets: An Experimental Study".

[2] Y. B. P. JungHyen An, "Methodology for Automatic Ontology Generation Using Database Schema Information," *Mobile Information Systems,* vol. 2018, p. 13, 2018.

[3] X. Tao, Z. Tang, C. Xu and Y. Wang, "Logical Labeling of Fixed Layout PDF Documents Using Multiple Contexts," in *11th IAPR International Workshop on Document Analysis Systems*, 2014.

[4] J. Chung and T. Delteil, "A Computationally Efficient Pipeline Approach to Full Page Offline Handwritten Text Recognition," in *International Conference on Document Analysis and Recognition Workshops (ICDARW)*, 2019.

[5] W. Li, W. Li and Y. Wu, "A Unified Model for Document-Based Question Answering," in *The Thirty-Second AAAI Conference on Artificial Intelligence (AAAI-18)*.

[6] T. Saba, A. S. Almazyad and A. Rehman, "Language Independent Rule Based Classifcation of Printed & Handwritten Text (Classication ofPrinted & Handwritten Text)," in *2015 IEEE International Conference on Evolving and Adaptive Intelligent Systems (EAIS)*, Douai, France, 2015.

[7] R. Karpinski and A. Bela¨ıd, "Combination of Structural and Factual Descriptors for Document Stream Segmentation," in *12th IAPR Workshop on Document Analysis Systems*, 2016.

[8] T. T. H. Nguyen, A. Doucet and M. Custaty, "Enhancing Table of Contents Extraction by System Aggregation," in *14th IAPR International Conference on Document Analysis and Recognition*, 2017.

[9] X. Yang, E. Yumer, P. Asente, M. Kraley, D. Kifer and C. L. Giles, "Learning to Extract Semantic Structure from Documents Using Multimodal Fully Convolutional Neural Networks," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

[10] M. K. Elhadad, K. M. Badran and G. I. Salama, "A Novel Approach for Ontology-based Dimensionality 5eduction for Web Text Document Classification," in *IEEE ICIS 2017*, Wuhan, China, 2017.

[11] M. Marrero, S. Sánchez-Cuadrado, J. M. Lara and G. Andreadakis, "Evaluation of Named Entity Extraction Systems," in *Advances in Computational Linguistics Research in Computing Science*, 2009, pp. 47-58.

[12] M. Marrero, J. Urbano, S. Sánchez-Cuadrado, J. Morato and M. Gómez-Berbís, "Named Entity Recognition: Fallacies, Challenges and Opportunities".

[13] B. W. Locke, "Named entity recognition: adapting to microblogging," in *Computer Science Undergraduate Contributions*, 2009, p. 29.

[14] A. Toral and R. Munoz, "A proposal to automatically build and maintain gazetteers for Named Entity Recognition by using Wikipedia".

[15] S. Jayalakshmi and D. Sheshasaayee, "Automated Question Answering System Using Ontology and Semantic Role," in *International Conference on Innovative Mechanisms for Industry Applications (ICIMIA 2017)*, 2017.

[16] V. G., D. Gupta, A. S. and S. Shaji, "A Graph-Based Relation Extraction Method for Question Answering System," 2017.

[17] Devlin, Jacob, Chang, Ming-Wei, Lee, Kenton, Toutanova and Kristina, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[18] Peters, Matthew E., Neumann, Mark, Iyyer, Mohit and Gardner, Matt and Clark, Christopher and Lee, Kenton and Zettlemoyer and Luke, "Deep contextualized word representations," in *Proc. of NAACL*, 2018.

[19] "Kaggle," [Online]. Available: https://www.kaggle.com/docs/tpu. [Accessed 20 03 2020].

[20] "TensorFlow Features | Why TensorFlow Is So Popular," [Online]. Available: https://data-flair.training/blogs/tensorflow-features/. [Accessed 20 03 2020].

[21] "Various Uses of TensorFlow," [Online]. Available: https://dzone.com/articles/learn-various-uses-of-tensorflow#:~:text=Other%20major%20TensorFlow%20Applications%20include,Self%20Driving%20Cars. [Accessed 20 04 2020].

[22] "TensorFlow 2.0 is now available!," [Online]. Available: https://blog.tensorflow.org/2019/09/tensorflow-20-is-now-available.html#:~:text=Today%2C%20we're%20delighted%20to,TensorFlow%202.0%20is%20now%20available!&text=TensorFlow%202.0%20provides%20a%20comprehensive,build%20scalable%20ML%2Dpowered%20applications.. [Accessed 01 05 2020].

[23] J. Alammar, "The Illustrated Transformer," [Online]. Available: http://jalammar.github.io/illustrated-transformer/. [Accessed 30 05 2020].

[24] "Transformers (State-of-the-art Natural Language Processing)," [Online]. Available: https://towardsdatascience.com/transformers-state-of-the-art-natural-language-processing-1d84c4c7462b. [Accessed 30 09 2020].

[25] J. Taylor, "ELMo: Contextual language embedding," [Online]. Available: https://towardsdatascience.com/elmo-contextual-language-embedding-335de2268604. [Accessed 04 01 2020].

[26] E. Ma, "ELMo helps to further improve your sentence embeddings," 30 10 2018. [Online]. Available: https://towardsdatascience.com/elmo-helps-to-further-improve-your-word-embeddings-c6ed2c9df95f.

[27] S. Vajpayee, "Understanding BERT — (Bidirectional Encoder Representations from Transformers)," [Online]. Available: https://towardsdatascience.com/understanding-bert-bidirectional-encoder-representations-from-transformers-45ee6cd51eef. [Accessed 23 08 2020].

[28] J. Alammar, "The Illustrated BERT, ELMo, and co. (How NLP Cracked Transfer Learning)," [Online]. Available: http://jalammar.github.io/illustrated-bert/. [Accessed 30 08 2020].

[29] R. Horev, "BERT Explained: State of the art language model for NLP," [Online]. Available: https://towardsdatascience.com/bert-explained-state-of-the-art-language-model-for-nlp-f8b21a9b6270. [Accessed 03 03 2020].

[30] Yiheng Xu, Minghao Li, Lei Cui, Shaohan Huang, Furu Wei and Ming Zhou, *LayoutLM: Pre-training of Text and Layout for Document Image Understanding,* 2019.

[31] J. Walter, *receipts,* kaggle.com.

[32] Guillaume Jaume, Hazim Kemal Ekenel and Jean-Philippe Thiran, "FUNSD: A Dataset for Form Understanding in Noisy Scanned Documents," 2019.

[33] guillaumejaume, "FUNSD: Form Understanding in Noisy Scanned Documents," [Online]. Available: https://guillaumejaume.github.io/FUNSD/. [Accessed 01 07 2020].

[34] Bowman, Samuel R. and Angeli, Gabor and Potts, Christopher, Manning and Christopher D., "A large annotated corpus for learning natural language inference," in *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2015.

[35] vishakha, *med_train.*

[36] A. Walia, *entity-annotated-corpus.*

[37] Jawahar, Minesh Mathew, Dimosthenis Karatzas and R. Manmatha, *{DocVQA}: A Dataset for VQA on Document Images,* 2020.

[38] "Overview - Document Visual Question Answering," [Online]. Available: https://rrc.cvc.uab.es/?ch=17.

[39] P. Rajpurkar, J. Zhang and K. Lopyrev, " SQuAD: 100,000+ questions for machine comprehension of text," in *EMNLP*, 2016.

[40] "CVPR2020 Workshop on Text and Documents in the Deep Learning Era," [Online]. Available: https://cvpr2020text.wordpress.com/.

[41] "google-research/bert," [Online]. Available: https://github.com/google-research/bert.

[42] t. f. e. Wikipedia, "F-score," [Online]. Available: https://en.wikipedia.org/wiki/F-score.

[43] "Research Group on Computer Vision and Artificial Intelligence," [Online]. Available: http://www.fki.inf.unibe.ch/databases.

[44] "neuromorphic computing," [Online]. Available: https://searchenterpriseai.techtarget.com/definition/neuromorphic-computing. [Accessed 06 09 2020].

[45] N. K. M. &. S. M. Kumar, "Automated ontology generation from a plain text using statistical and NLP techniques," *Int J Syst Assur Eng Manag*, vol. 7, p. 282–293 , 2016.

[46] D. S. Jeong, "Neuromorphic spiking neural networks for temporal learning," Hanyang University, Seoul.

[47] J. Devlin, M.-w. Chang, K. Lee and K. Toutanova, *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding,* 2019.

[48] A. I. F. AI, *COVID-19 Open Research Dataset Challenge (CORD-19).*

[49] L.L. Wang, K. Lo, Y. Chandrasekhar, R. Reas, J. Yang, D. Eide, K. Funk, R.M. Kinney, Z. Liu, W. Merrill, P. Mooney, D. Murdick, D. Rishi, J. Sheehan, Z. Shen, B. Stilson, A. Wade, K. Wang, C. Wilhelm, B. Xie, D. Raymond and D. S. Weld, "CORD-19: The Covid-19 Open Research Dataset," *ArXiv*, 2020.