Knowledge Based Deep Inception Model for Web Page Classification

Amit Gupta* and Rajesh Bhatia

Department of Computer Science and Engineering, Punjab Engineering College (Deemed to be University), Chandigarh, India
E-mail: amitgupta.phdcse15@pec.edu.in; rbhatia@pec.edu.in
**Corresponding Author*

Received 11 March 2021; Accepted 22 July 2021;
Publication 22 October 2021

Abstract

Web Page Classification is decisive for information retrieval and management task and plays an imperative role for natural language processing (NLP) problems in web engineering. Traditional machine learning algorithms excerpt covet features from web pages whereas deep leaning algorithms crave features as the network goes deeper. Pre-trained models such as BERT attains remarkable achievement for text classification and continue to show state-of-the-art results. Knowledge Graphs can provide rich structured factual information for better language modelling and representation. In this study, we proposed an ensemble Knowledge Based Deep Inception (KBDI) approach for web page classification by learning bidirectional contextual representation using pre-trained BERT incorporating Knowledge Graph embeddings and fine-tune the target task by applying Deep Inception network utilizing parallel multi-scale semantics. Proposed ensemble evaluates the efficacy of fusing domain specific knowledge embeddings with the pre-trained BERT model. Experimental interpretation exhibit that the proposed BERT fused KBDI model outperforms benchmark baselines and achieve better performance in contrast to other conventional approaches evaluated on web page classification datasets.

Journal of Web Engineering, Vol. 20_7, 2131–2168.
doi: 10.13052/jwe1540-9589.2075

Keywords: Web page classification, transfer learning, knowledge graph embedding, pre-trained model.

1 Introduction

Ever-increasing growth for the number of web pages on the Internet presents the governance and management of web pages a denounce task. Users are faced with problem of pertinent and nimble excerption for web pages and require enhanced web surfing and crawling experience. World Wide Web (WWW) has witnessed a flashing enlargement in terms of web content data related to various information domains, which becomes more decisive to fetch the required data for the expected result. Search engines can localize the information available on web but can't have ability to organize it well. Web Page Classification being pivotal for information management in aspect for creating web repositories and for information retrieval in aspect for listing classified directories in NLP domain problem.

Assigning predefined labels for classifying web pages using a pertinent subset of relevant features from web pages to predict the category of web page refers to the process of Web Page Classification. Classifying web pages in an automated aspect [42] is an analytical task for facilitating improved and efficient retrieval of web pages related to different domains.

Pre-trained model trained on large text corpus as in transfer learning based models can be re-purposed on a downstream related problem during fine-tune the target task. Pre-trained models like deep-bidirectional BERT [37] using transfer learning approach achieves state of the art result for text classification problem. BERT learns contextual semantic representations for input tokens using bi-directional attention in different transformer encoder layers.

Feature-based [43] and fine-tuning-based [36] language models trained on large text corpus have ability to capture rich semantic information from the text and can be further fine-tuned for the purpose of task specific downstream tasks in NLP domain. Pre-trained language models based on Transformer Networks like BERT continue to show state-of-the-art results on a wide variety of NLP processing tasks. BERT model uses the bi-directional attention mechanism in order to learn the contextual meaningful information from within a sentence to excerpt the meanings.

Knowledge Graphs [39] can provide rich factual knowledge representations for language modelling task. Knowledge embeddings from structured graphs can be used to optimize the local interactions among language token vocabulary in order to learn valuable representation.

We proposed an ensemble Knowledge Based Deep Inception (KBDI) model for fusing domain specific knowledge embeddings with the pre-trained BERT model. The BERT fused KBDI model not only highlights the limitations with BERT model in learning local vocabulary dependency information but also demonstrates how to incorporate the domain knowledge to enhance the word embeddings produced by BERT. This useful domain knowledge can used to complement the domain-specific NLP application by fusing knowledge representations into BERT model which makes BERT feasible for learning global information available in web pages.

For the first time in the proposed model we assess the potency of fusing rich factual knowledge information with BERT model for the purpose of classifying web pages into different categories. A novel ensemble approach for Web page classification is proposed in which contextual semantic representations were learned using BERT model and then these embeddings are fused with extra domain-specific knowledge embeddings and then fine-tune the target task using Deep Inception network [50] employing labelled dataset in supervised learning fashion.

We used Deep Graph Knowledge Embedding Library (DGL-KE) to train and test the knowledge embeddings from structured graphs using an embeddings library built on top of the Deep Graph Library (DGL). Scalable and distributed Python library interface is used for excerpting knowledge embeddings from structured graphs using an easy-to-use and high-performance DGL library. We create knowledge embeddings from task-specific domain knowledge graph consists of entity nodes and relations and then validate those embeddings.

Pre-trained contextual word embeddings trained on large text corpus using bi-directional transformer based BERT model fused with knowledge richer embeddings from Knowledge Graphs are used along with deep inception [53] module. We endorse the fusion of knowledge information in order for better insight of local meanings in the input sequence and in learning representations with global meanings.

In summary, the main contributions of this paper are as follows:

1. How fusion of domain specific knowledge graph embeddings with pre-trained models can empower the classification of web page into pre-defined categories.
2. Fine-tuning target task using Deep Inception network on top of the pre-trained model can learn discriminative representation of input embeddings at multi-level semantics.

3. The proposed model accuracy resulting from ensemble knowledge embeddings with BERT along with structured deep inception module can scores well against other standard baseline classification approaches.

2 Web Page Classification

Different researchers have presented various approaches in the domain of web page classification. Number of machine learning techniques and algorithms which are widely used by different researchers are elaborated in this section. Research work [1–3] was performed in the literature for information management and retrieval domain. Distributed crawling and fetching of web pages as in [4] to discover relevant web pages was explored to highlights scalability issues in approaches.

Web Page Classification problem can be performed in automated manner by training classifiers using machine learning methods. Abounding studies matured in the literature for web page classification [5–7] using Supervised and Unsupervised Machine Learning approaches.

Traditional web page classification algorithms using machine learning techniques include K-NN [10] approach, Bayesian probabilistic models [12] and Logistic Regression. Algorithms like Support Vector Machines (SVM) [8, 14], decision trees and Neural Networks [11] are used for the classification task. Different traditional algorithms use relevant feature sets for training the classifier and extracting desired attributes. Feature selection methods either covered the feature sets or wrapper methods for the purpose of feature selection.

Metaheuristic approaches for relevant feature selection are explored using Genetic Algorithm [9, 15, 22] and Ant Colony algorithm [13] to optimize the feature subset search space. Ensemble modelling is performed to combine the effectiveness of multiple approaches and their combination is explored with different objectives. Feature selection using Naive Bayes algorithm is explored by authors in [6] to enhance the predictive performance of the approach using consistency based relevant selection of feature subsets during classification.

2.1 Deep Learning Techniques

Traditional machine learning classification techniques uses extraction of a relevant subset of features from the content of web pages in order to classify the web pages in predefined categories. Modern deep learning techniques

based on artificial neural networks (ANN's) combine's representational learning approach as in biological systems for processing of information and distributed communication between nodes. The "deep" describes the usage of multiple hidden layers in the artificial neural network in referring deep learning terminology.

Modern deep learning techniques learns weight vectors for different features while processing the input features through multiple deep hidden layers as compared to conventional approaches. Usage of deep hidden layers provides the ability to grasp abstract features through the hidden layers in an accelerated manner. Each layer passes learned weight vectors to the later layers for processing of information.

For text classification problem deep learning techniques like Convolutional Neural Networks (CNN) [22], Recurrent Neural Networks (RNN) [21] and ensemble techniques are studied in literature. Models using CNN [19, 20] applies Convolutional filters to the input text using multiple filters to obtain localized information embedded in the input text.

Neural models based on RNN [26] can sustain the limitation of ANNs by storing the long-term information in the input sequence. RNN catch long-term dependencies and models the complete given sequence but it lags in finding key patterns in input representation, CNN on the other hand does well in extracting local and finding position-invariant features in input representation.

CNN [30] and RNN based ensemble methods are used to model neural methods like C-LSTM [25], CNN-LSTM [27, 28] and DRNN [23, 48]. C-LSTM uses the CNN to apply Convolutional filter for capturing the input text localized information and then combines it with LSTM to get more global input textual representations.

DCNN [24, 46] uses dynamic pooling using max-K tokens at a time to capture the features having maximum feature values in input text. Disconnected recurrent neural network (DRNN), assimilate position-invariance feature vectors into RNN, at every time stride of the model the hidden state is confined to show up words neighbouring the current position vector by limiting the information flow distance.

To concern on a subset of the key information in the input sequence attention-based models adapt this capability. These attention-based models is part of sequence-to-sequence models which aims to use variable length input sequence and produces output sequence of different length using encoder and decoder mechanism. These models use RNN's for the purpose of encoding the initial sequence using Encoder and produces a progression

of hidden-states for every time-step. These hidden units served the purpose of 'memory' for these attention networks to store the intermediate state of the network and are used by decoder to produces the variable-length output sequence at different time-steps.

The HAN [29] applied two level of attention differently at sentence and word level to differentiate the important and non-important representation of content from the document in order to select the qualitative information.

Deep Inception Networks

To faster the learning process for deeper networks residual learning frameworks are used. Shortcut connections to deeper layers resemble the identity function which helps in vanishing gradient while algorithm learns in backward direction. ResNet [28] learns the residual functions for shortcut connections resembling identity function for skipping number of layers in deeper network. Inception [31, 32] uses deep convolutional neural network architecture with wider and dense network for stacking inception modules on top of each other. Inception-v4 [33] achieved better performance with conjunction of residual layers on deep convolutional layers. Inception model architecture with residual shortcut connections achieves good performance on classification task.

2.2 Pre-Trained Models for Text Classification

Model pre-trained for initial job as in transfer learning based deep learning models can be re-purposed on a different but related objectives achieved great performance while fine-tune the target task. Pre-trained models using transfer learning approach attain impressive performance for text classification. Attention-based models combined with word embeddings generated from deep networks are proven to be effective in language translation and NLP tasks.

Traditional Context-Free word embeddings have limitation in assuming to infer the stable meaning of a word in the input sequence which limits the polysemy in input tokens. The two most common context-free embeddings word2vec and Glove use shallow representations for the weight vectors generated from networks being shallow in deepness.

Pre-trained representations for word vectors removes the limitation of conventional context-free embeddings by learning contextual semantics for the input sequence from large amount of unlabelled dataset. Some of the effective transfer learning approaches used are Embedding from Language

Models (ELMO) [34], Universal Language Model Fine-Tuning (ULMFiT) [36], OpenAI' s Generative Pre-trained Transformer (GPT) [35], and Google's BERT [37] model using unsupervised approach for pre-training.

ELMO [34] learns word representations using shallow deep bidirectional language model with character convolutions to represent contextual enabled meanings of the input sentences. ULMFiT [36] captures the general feature by training the network on a broad dataset and learns features in different layers of the network and make tuned it on a objective task using discriminative analysis.

Transformer networks are models having attention network on top of pre-trained language representations. GPT [35] uses a multi-layer unidirectional decoder of transformer network model for representation of contextual embeddings and fine-tunes the model for downstream tasks. BERT [37] an unsupervised modelling, deep-bidirectional approach for pre-trained language representations disciplined on large unlabelled text corpus learns contextual representations for input sequences applying attention based mechanism in different transformer encoder layers to grasp context-based semantics of the input tokens. BERT uses two learning strategies including Masked Language Model (MLM) and Next Sentence Prediction (NSP) for purpose of language modelling employing unsupervised approach for learning representations of the input sequences.

2.3 Knowledge Graphs and Embeddings

Semantic knowledge embeddings where knowledge representation is crucial has emerged as standard for fusing semantic meanings into the world of Web Data. Numerous number of knowledge graphs have been suggested in literature ranging and encompassing from Google Knowledge Graph [39], Wikidata [18, 40], DBpedia [42], YAGO [41], Freebase [16], and NELL [17].

Task specific domain knowledge graphs which are contextual in the manner for knowledge representation such as Google Knowledge Graph, Wikidata, YAGO, Freebase, and NELL have comprised millions of real world entities like people, attributes and locations with number of inherited relations exists between them. In literature work exercise was created to intensify task specific domain knowledge graphs to incorporate domain vocabulary in process for creating knowledge base having structured factual relationships.

Freebase and Wikidata knowledge base were created by collaboration of users for purpose of maintaining structured data and its representation. Semi-structured information resources like info boxes and links pointing to

external world are used by DBpedia and YAGO in process for their creation. NELL knowledge base was in charge of inhabiting a base with several knowledge beliefs to be scratched from web pages and text corpus.

Knowledge graph embeddings provides a good enriched, high quality, structured knowledge representations to enhance pre-trained word embeddings by incorporating task-specific embeddings in order for assimilating extra morphological, syntactic and semantic representations. Knowledge graph embeddings are representing low-dimensional vector space depiction for the entities and relations exists in real-world to be incorporated in a knowledge graph. They hypothesize semantic representations for the given entity node using its structured relations.

Popular knowledge graph embedding models exists in literature, including TransE, TransR, RESCAL, DistMult, ComplEx, and RotatE. Each different embedding model has a varying score function which is used to compute the distance between two similar entities by analyzing semantic relations exist between them. Entities are more closer in vector space if there exists a semantic relation between them, else if there is no relation between them they are far in vector dimensions.

Deep Graph Knowledge Embedding Library (DGL-KE), a knowledge graph embeddings library built on top of the Deep Graph Library (DGL), is a library which is having easy-to-use interface, having high performance which can run on CPU or GPU machine with scalable architecture for representing large-scale knowledge graph embeddings. We used DGL-KE for computing the low dimensional embeddings for representation of entities and relations on top of the domain specific knowledge graph.

DGL-KE is designed with perspective for training and testing knowledge graph embeddings at large scale which can run on cluster of machines. As there are millions of entity nodes with billions of semantic relations between them in domain specific knowledge graph, it requires to train these embeddings with efficient optimized architecture that accelerate the training process using multi GPU parallelism and less memory overhead.

3 Proposed Methodology

Exploiting knowledge graphs in an analytical and orderly manner could provide well defined source of kind human crafted knowledge information for culturing superior embedding related to specific tasks at hand. Language models trained on large text corpus can learn bidirectional contextualized

semantic meanings using a supervised training fashion to show state-of-the-art results on natural language problems.

To enhance pre-trained word embeddings available from pre-trained models proposed study assimilates information related to task specific domains in order to fuse extra knowledge aware representations to expedite the classification of web pages. Fusion of knowledge information entitles the classification of web pages; by constructing a domain knowledge graph related to task domain. Knowledge embeddings curated from structured knowledge graphs can be applied to related downstream application tasks.

For enhancing the embedding representations for the input tokens applying knowledge information can be operated either by jointly fusing both word and graph embeddings simultaneously or post fusion of trained knowledge representations directly into pre-trained language models. We concentrate on evaluating the post fusion scenario by incorporating knowledge embeddings directly into pre-trained BERT to classify web pages.

Automated fashion of constructing knowledge graph from unstructured text for task specific applications presents a confronting situation from years; here we described a detailed surround for the steps involved for the purpose of classifying web pages into pre-defined categories, initiating from acquiring knowledge ranging to constructing task specific domain knowledge graph and later fusing factual knowledge information into BERT model trained on large text corpus.

We employ Deep Inception module on top of pre-trained BERT model, to fine-tune the factual knowledge fused embeddings for the target domain based task of classifying web pages into different categories. Contextual pre-trained embeddings are extracted from twelve encoder layers of BERT model with bi-directional depiction which are then fused with factual syntactic knowledge embeddings and then the target task is fine-tuned by exploiting a Deep Inception module to update the weight vectors using available defined classes' dataset for web pages.

The model architecture is visualized in Figure 1. For acquiring knowledge information, list of web page URL's is served as input to the knowledge acquiring module. The URL's are then parsed with pre-processing techniques for the purpose of crawling web page data and then transformation step is used for converting data into native serialized binary format in tensorflow framework for the objective of optimizing the data pipeline during the input flow process. Detailed steps for the above process are explained in Section 3.1. Token embeddings from input web pages extraction process

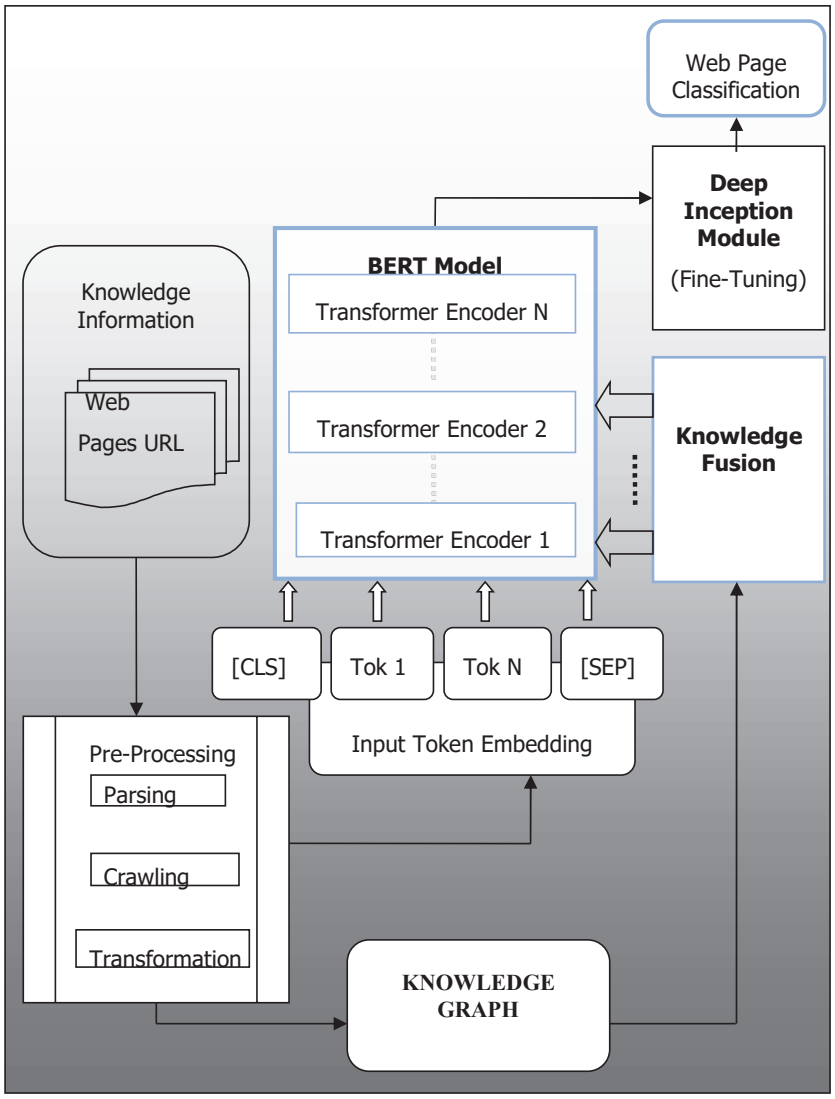

Figure 1 Proposed BERT fused KBDI model architecture.

and there flow to BERT model is discussed in Section 3.2, and the process for complementing these embeddings with factual domain task specific knowledge embeddings is explained in Section 3.3.

As the input parameter length being a crucial factor in pre-trained models, we input a maximum 256 tokens as input from web page content to

pre-trained BERT model for extracting contextual bi-directional embeddings of the input tokens in hidden dimensional space of 768 units. BERT specifically uses two special tokens as input which are [CLS] and [SEP] to mark the start and end of the input sequence to indicate the starting point and last of the input token sequence.

3.1 Knowledge Information Acquisition

The process for collecting pragmatic knowledge information from various sources in course to design the base for knowledge establishment in structured graph is referred as knowledge acquisition process. Information format differs among different knowledge base systems, varies in terms of text structure, web pages or in format of databases. This inconsistency among the stored information arrangement needs to be solved for efficient collection of useful information.

For generating useful knowledge and structuring the knowledge bases for the objective of constructing a domain task-related knowledge graph, we concentrate on representing knowledge information in a structured formatted manner. We progressed with bottom-up trend for knowledge annotation and collect web pages relevant to ten different domains available in DMOZ open source dataset.

Knowledge Base Dataset

DMOZ dataset is used for constructing structured knowledge graph. DMOZ dataset is an open source URL classification dataset for web pages from dmoz directory. For the purpose of classifying web pages we used ten most popular domains of web pages from dmoz directory including Shopping, Health, Business, Science, Computers, Recreation, Games, Arts, Business, and Society. World Wide Web (WWW) links for different web pages from different domains are included in dmoz dataset. File content.rdf.U8.gz of size 238 MB needs to be downloaded from ODP and to be extracted using gzip compression to excerpt content.rdf.U8 file of size 1.57GB. File content.rdf.U8 is a Resource Description Framework (RDF) resource file encoded in UTF-8 encoding as shown in Figure 2.

Parsing and Crawling of Web Pages From DMOZ Dataset

XML parser is used to parse the content from the content.rdf.U8 file using xml.sax.make_parser () method to extract the dmoz_10 file. The parsed

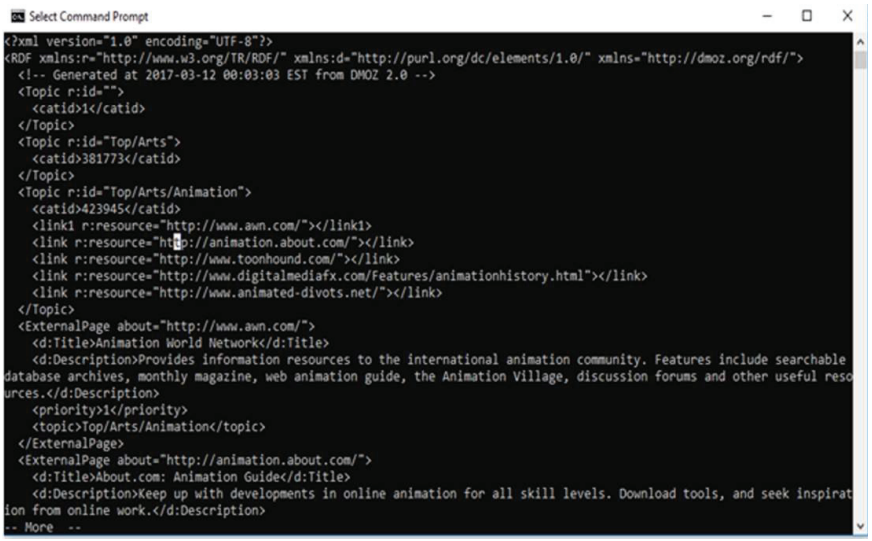

Figure 2 RDF File for DMOZ dataset.

dmoz_10 file contains URL's for different categories. 41899 for Health, 71762 for Sports, 171693 URL's for Business, 78977 for Computers, 169029 for Society, 69656 for Recreation, 164850 for Arts, 60885 for Shopping, 28256 for Games, and 79719 URL's for Science categories are available in dmoz_10. Extracted URL's for Business domain after parsing are shown in Figure 3.

For fetching and crawling web content we used urllib.request.urlopen ([“url”]) method inside a python dictionary variable. Different domains are crawled separately and fetched content are stored individually in different files. Science category web page content after crawling is shown in Figure 4. Nested scope is created in python list variable for creating different dictionaries to stock the crawled “html” web page content with its “url” for web page and unique “id” identifying variable.

Fetch-timeout of 20 seconds max is used during crawling web page content for different web pages. Multithreading is used with ten different numbers of threads in synchronous mode used to fetch data with different HTML tags in web pages. Nine lakh thirty six thousand seven hundred twenty six (9,36,726) web page URL's are crawled for fetching their content regularly for 17 days in desire to create knowledge base for constructing task specific domain semantic knowledge graph. This knowledge base information is required during data input pipeline processing.

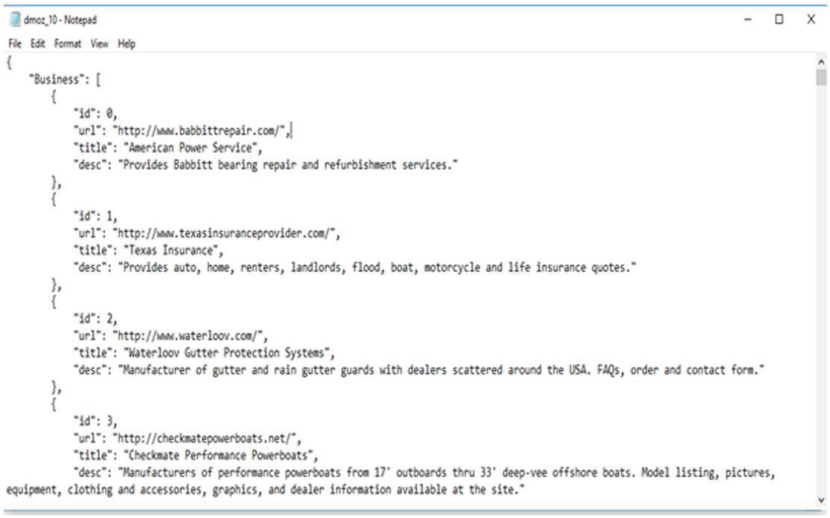

Figure 3 Parsed dmoz_10 content file.

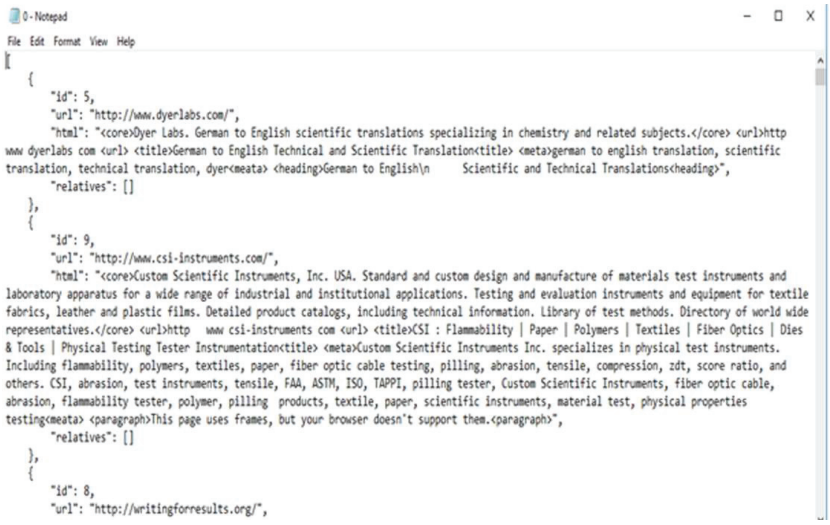

Figure 4 Crawled data information.

3.2 Constructing Task Specific Domain Knowledge Graph

Constructing knowledge graph related to task specific domain from data available in unstructured format represents a challenging problem. This problem is addressesed in an automated fashion by extracting named entities

and their different attributes from web pages, and models the relationships between them. Knowledge triplets in form of "entity-relation-entity" pairs extracted from unstructured data formulates an element for constructing the domain knowledge graph. These plucked knowledge triplets are later used by graph database to construct the semantic graph.

Knowledge graph construction comprises two key processes "Information Extraction" and "Knowledge Fusion". Information extraction composed of Named Entity Recognition for sever named entities with their data attributes and Relation Extraction in order to infer relations manifest among different named entities. Co-reference resolution is performed during knowledge fusion phase to finest the aspect of knowledge graph.

A standard pipeline-based model is examined for extracting the knowledge triplets. Various deep learning algorithms were tested and evaluated during different phases for constructing domain knowledge graph. We examined different python script algorithm described in ALGOL below to evaluate the process of entity recognition and relation extraction on domain-specific data.

Steps performed during constructing domain knowledge graph are outlined in Figure 5 below. We excerpts the web content having fruitful html tags from web pages and performed afterwards text pre-processing, removing unwanted characters and variables. Different steps for constructing knowledge graph are outlined below.

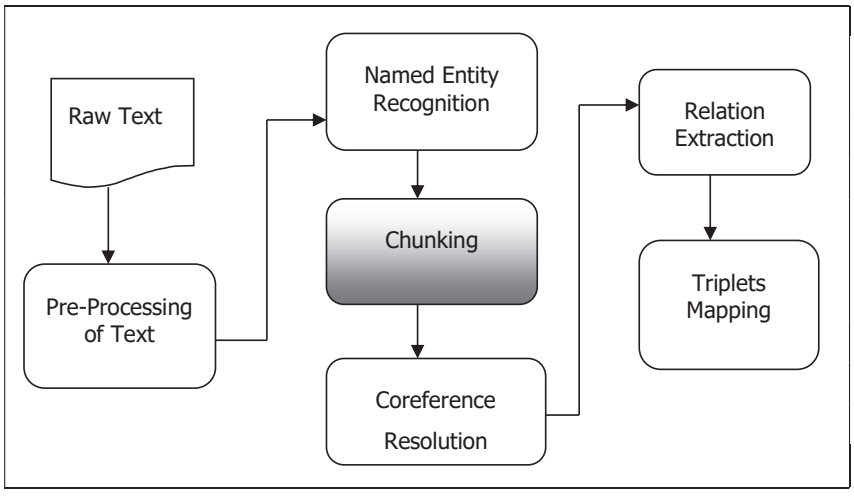

Figure 5 Pipeline model.

Named Entity Recognition and Chunking

Named Entity recognition points to process for discerning named entity mentions related to person, places, organizations with their attributes like age, size in addition to facts revealing to entities evinced in available data.

Different natural language processing tools like Stanford NER, NLTK, and SpaCy are evaluated for excerpting entities from available data. Evaluation criteria are based on creating confusion matrix for different tools and compute the precision for different approach and then select the best processing tool.

The unstructured data available is processed for removal of special characters and then split across different sentences for tokenizing it into input word tokens to be tagged with Part of Speech (POS) tag with their dependency parsing information. Table 1 points the named entity recognition

Table 1 Dependency parsing and named entity recognition

Id	Token	Entity Type	IOB	CoarseGrained POS	POS	Dependency
0	Andalusia	ORG	B	PROPN	NNP	compound
1	Health	ORG	I	PROPN	NNP	compound
2	medical	ORG	I	PROPN	NNP	nsubj
3	includes		O	VERB	VBZ	ROOT
4	an		O	DET	DT	det
5	crossed	NORP	B	ADJ	JJ	amod
6	tour		O	ADJ	JJ	amod
7	Country		O	NOUN	NN	attr
8	bulletin		O	DET	WDT	nsubj
9	programs		O	VERB	VBZ	relcl
10	patient		O	DET	PRP	poss
11	and		O	ADJ	JJ	amod
12	employment		O	NOUN	NN	dobj
13	center		O	ADP	IN	prep
14	news	GPE	B	PROPN	NNP	pobj
15	hospital		O	PUNCT		punct
16	services	GPE	B	PROPN	NNP	appos
17	emergency		O	PUNCT		punct
18	room		O	DET	DT	det
19	health		O	NOUN	NN	dobj
20	care		O	ADP	IN	prep

Table 2 Chunking rules for named entities

Noun and Verb phrase chunking

```
Input: web page documents
Output: chunked_entities
    Function Noun_Verb_Chunking(input_document)
    BEGIN
        noun_phrase_rule := {<DT|PRP\$>?<JJ.*>*<NN.*>+}'
        for sentence ∈ web_document do
            BEGIN
                noun_phrase := Parse noun phrases and chunk them using noun_phrase_rule'
            END'
            for sentence ∈ web_document do
                BEGIN
                    verb_phrase := Parsing verb phrases and tag them as VERB'
                END'
                chunk_list := [ ]
                Append noun phrases and verb phrases in chunk_list
          return chunk_list
    END'
```

with their dependency parsing information along with entity id and IOB in health domain web page from DMOZ dataset.

Named entities are chunked for excerpting relevant concern sentence phrases from the input. Part of Speech (POS) tags are used with grammar rules for evaluating entities that can be chunked. Different chunked entities will behave as single named entity as equivalent to binary = true option. Table 2 highlights the chunking rules in ALGOL.

Entity Coreference Resolution and Relation Extraction

Coreference Resolution points to the process for finding linguistic expressions among named entities that refers to the same entity during inference phase. ALGOL code for coreference resolution is elaborated in Table 3 to find out coreferences among named entities using Stanford Core NLP tool. Different web pages named entities are linked to multiple references in knowledge base which needs to be resolute for efficient extraction of relations among entities.

Extraction of binary relations from unstructured text for excerpt semantic relationships in form of knowledge triplets refers to relation extraction process. Knowledge triplets in form of "Entity1 – Relation – Entity2 are excerpted from named entities". Relationships among named entities are

Table 3 Coreference resolution for entities

```
Coreference Resolution
Input: web page documents
Output: corefernce resolved
    Function generate_coreferences(input_document)
    BEGIN
        linguistic_forms := Call StanfordCoreNLP to generate lingusitic forms'
        properties := select properties using "coref" annotators'
        annotation := input_document annotation using above properties'
        output := load json using above annotation'
    END'
    Function resolve_coreferences(input_document, named_entities, generated_coref)
    BEGIN
        corefernces := generated_coref from function generate_coreferences()'
        sentence_wise_replacements := defaultdict(list)'
        for index,coref ∈ enumerate(corefernces.values()) do
            BEGIN
                # replace corefernce by finding their reference
                for ref in coref do
                    if ref in enlity_list.keys() then
                        BEGIN
                            replace_ref with ref
                        END'
            END'
            ref_list := [ ]
            Append replace_ref to ref_list
    END'
```

extracted for pointing out the relation edges and to find out the entity nodes in knowledge base. ALGOL code for excerpting relations among entity nodes is elaborated in Table 4.

3.3 Knowledge Fusion

For fusing the knowledge representations into BERT model which is already trained on large text corpus, we refers to the tokens from web pages which are similar with the corresponding named entities in the task specific domain knowledge graph, we points them'entity mentions'. Web page with m input tokens, here we consider maximum value of m equals to 256 for the input tokens, i.e. $\mathrm{W} = \{w_i\}_{i=1}^{m}$, have to be classified into web pages into c different unique categories, where maximum value of c is 10. Input web page with m tokens is first formed into a input sequence of length $m + 2$, by inserting two

Table 4 Relation extraction among named entities

Relation Extraction

Input: web page document
Output: relation triplets
Function standford_openie(input_filename)
BEGIN
args:=java -mx1g -cp "*" edu.stanford.nlp.naturalli.OpenIE /path//input_filename
create_child_process = Call class subprocess.Popen(args)'
raw_entity_relations = read_entity_relations(create_child_ process)'
results = generate_graphviz_graph(raw_entity_relations)'
END'
Function relation_extractor(named_entity_files)
BEGIN
entity_relations = { }'
for ne_file ∈ named_entity_files **do**
BEGIN
relations = Call standford_openie(ne_file)'
entity_relations = append(relations)'
END'
END'

BERT special tokens [CLS] and [SEP] at the very start and end of each input token sequence, i.e. input token sequence S = [<CLS>, W, <SEP>].

Input sequence of word tokens are represented as accumulation of token, position and segment embeddings for each w_i in S.

$$R_l^i = w_i^{token} + w_i^{position} + w_i^{segment} \tag{1}$$

Transformer encoder layer transforms the input sequence as in (1) by passing it through different self-attention layers.

$$R_l^i = \textit{Transformer\_Blocks}(R_l^{i-1}),\ \mathrm{i} = 1, 2, 3, \ldots, \mathrm{N} \tag{2}$$

BERT with N (N = 12) layers of encoder blocks in bidirectional mode is used for extracting contextual representations with each transformer layer computes a multihead self-attention values for query vectors Q, key vectors K and value vectors V. Attention score for transformer layers are evaluated using (3), where $\sqrt{d_k}$ is used as a scaling factor .

$$\textit{Attention(Q, K, V)} = \textit{Softmax}(QK^T/\sqrt{d_k})\mathrm{V} \tag{3}$$

Each input token w_i evaluates a weighted vector representation using the attention scores as computed in (3) for encoding the textual contextual information from its neighbouring tokens from each web page.

Post Training Knowledge Fusion

BILINEAR model explained in [38] is used in proposed model for computing the candidate score for each entity concepts from concerned triplets (e_1,*r*,e_2) via a bilinear function used, where each candidate entity concept e_i is accomplice with continuous vector embedding V_{e_i}:

$$\mathit{SCORE}(e_1, r, e_2) = V_{e_1}^T M\ V_{e_2} \tag{4}$$

Where V_{e_1} and V_{e_2} are vector embeddings for entities e_1 and e_2, and *M* is a relation-specific embedding matrix. Entity concept e_i is computed for relevance via a attention weight operation A_w:

$$A_w \propto \mathit{exp}(V_{e_i}^T\ W\ R_i^l) \tag{5}$$

Web page category c at the final layer N of the transformer encoder model is predicted on the base of corresponding contextual representation R_i^l:

$$p(y_i|\boldsymbol{c}) = \mathit{softmax}(R_i^l|\boldsymbol{c}), \tag{6}$$

$$L = \sum \mathit{log} p(y_i|\boldsymbol{c}) \tag{7}$$

Negative sampling is employed randomly during the training process as:

$$L_{\mathit{Neg}} = \sum \mathit{log}\ p(y_i|\boldsymbol{c}) \tag{8}$$

Where y_i is a negative sample which is randomly elected.

Final objective of post-training for knowledge fusion, where δ is a coefficient is:

$$L_{\mathit{Post}} = L + \delta \cdot L_{\mathit{Neg}}, \tag{9}$$

Pre-Trained BERT Model

We exploit the contextual embeddings obtained from 12 transformer encoder layers of BERT Base model having 12 different attention heads in each encoder layer. Each different head in separate encoder block utilize self-attention mechanism for generating contextual embeddings of the input tokens. Sequence length of maximum 256 input tokens from individual web pages are used as input to pre-trained bidirectional BERT model. Contextual embeddings of the input tokens using BERT are extracted in 768 hidden dimensional output vectors. Two special tokens [CLS] and [SEP] are inserted

by BERT at the start and end of the input sequence to mark the beginning and finish of the input sequence.

Input tokens of sequence length 256 in batch size of 16 are used. The input tokens of shape [16*256] are used to generate input embeddings having maximum embedding length of size 768 with shape [16*256*768]. These input embeddings are combined with token-type embeddings of shape [16*256*768] and positional encodings of shape [16*256*768] to generate the final input embeddings of shape [16*256*768] which are provided to initial layer of BERT Base model. Each layer produces an output of shape [16*256*768]. BERT uses the final layer output of shape [16*256*768] and utilizes the pooler to transform the final layer output to shape [16*768]. BERT assumes the embedding of [CLS] token has been pre-trained and uses it as the representation of the whole input sequence of tokens. We use the BERT Base model to classify web pages into 10 different categories: Business, Society, Science, Recreation, Shopping, Games, Arts, Business, Computers and Health.

3.4 Fine-Tuning Process

For fine-tune the target task, in spite of using BERT Base model which projected the final layer output and apply softmax activation on the projected layer output, we used BERT last encoder layer output of shape [16*256*768] and passed it over to Deep Inception module for classification of web pages into pre-defined categories.

Architecture for deep inception model for fine-tuning is explained in Figure 6. The BERT model last layer output is initially convolved with three convolution layers for inferring deeper insights with different filter sizes and multiple filters in order to focus on the contextual representation of the local information embedded in the input tokens. BERT final layer output is expanded to [16*256*768*1] where last dimension '1' represents the number of channels.

Three different convolution layers CONV 1, CONV 2 and CONV 3 are used each having varying filter size of [2*768], [3*768] and [5*768] dimension with 32, 64 and 64 filters to be convolved with input of size [16*256*768*1]. Residual connection from input of inception module to the output of CONV 3 layer is used to make the dimensional match. We used a shortcut connection using CONV layer with [5*768] filter size with 64 filters.

Three convolution layer computations are performed on input to excerpt the local information. Batch Normalization is activated on the channel axis

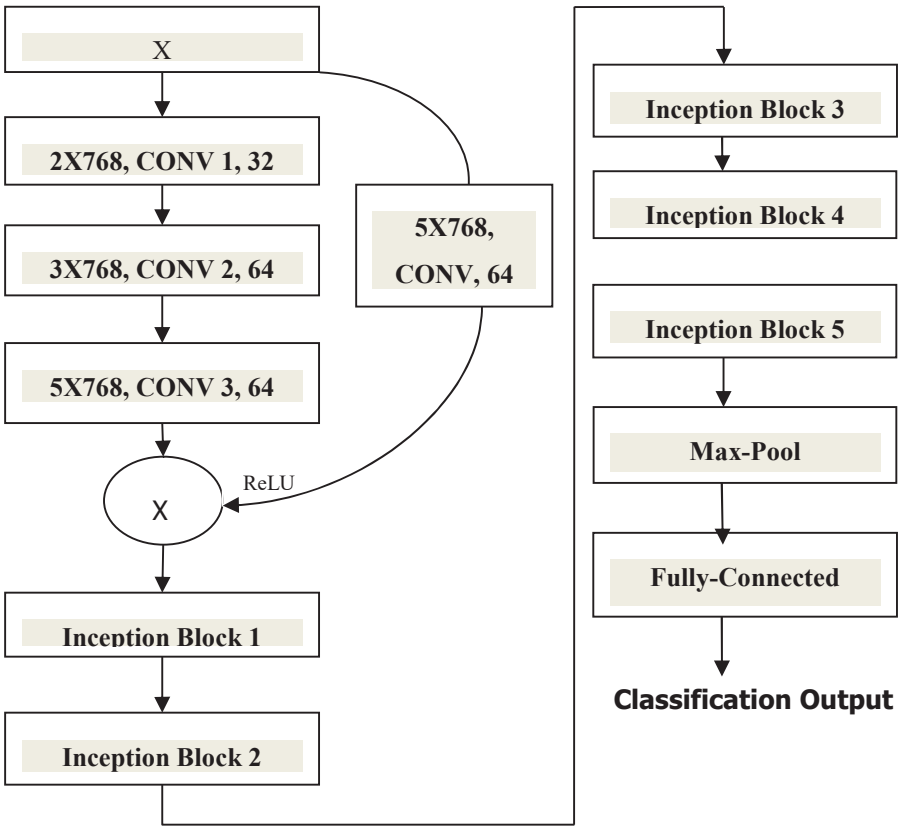

Figure 6 Inception module.

to evaluate with higher learning rate and then Rectified Liner Unit (ReLU) activation is used to add non-linearity to the model hyper parameters. Five inception blocks are used after convolution layers to extract multi-scale contextual features.

Max-Pool layer is used to diminish the size of the input matrix after applying inception blocks in deep inception module. A fully connected dense layer of size 128 neurons is applied on max-pool layer output to be followed by Softmax activation.

Different blocks of inception are used to extract multi-scale semantics using the convolutional operation with different sized kernel with varying numbers of filters. Each inception block layer details are mentioned in Table 5.

Table 5 Deep inception module layer

Layer	No. of Filters	Filter Shape	Output Shape
Previous Activation (X)	–	–	$256 \times 768 \times 1$
Conv (X)	64	5×768	$252 \times 1 \times 64$
Conv 1	32	2×768	$255 \times 1 \times 32$
Conv 2	64	3×768	$253 \times 1 \times 64$
Conv 3	64	5×768	$249 \times 1 \times 64$
RELU (Conv(X) + Conv 3)	–	–	$249 \times 1 \times 64$
Inception Block 1	16	Multiple filters	$122 \times 1 \times 96$
Inception Block 2	32	Multiple filters	$122 \times 1 \times 128$
Inception Block 3	32	Multiple filters	$122 \times 1 \times 192$
Inception Block 4	64	Multiple filters	$122 \times 1 \times 224$
Inception Block 5	128	Multiple filters	$122 \times 1 \times 256$
Max-Pool Layer			$1 \times 1 \times 768$
Max-Pool Reshape			768
Fully-Connected Layer			128
Softmax layer			10

We learn more distant dependencies inside input token sequences using combination of [1*1], [3*3] and [5*5] convolution operations to facilitate higher learning rate in reaching global loss during backward propagation. Hyper-parameters are optimized in inception blocks using non linear activations along with CONV layers.

In inception module each block is having six parallel multi-branches with different convolution kernel of sizes, i.e. 1×1, 3×3 and 5×5 are used inside the blocks to convolve the input matrix in the block to excerpt the contextual features at multiple level scale using varying sized kernels.

4 Experiments and Results Evaluation

For the objective of web page classification different datasets were studied including datasets used in normal text classification.

4.1 Datasets Crawled

Benchmark datasets which are crawled and transformed for the purpose of web page classification are outlined and briefed based on size of train and test examples along with classes as in Table 6.

Table 6 Benchmark datasets evaluated for web page classification

Datasets	Dataset Description and Size	Train (Labelled) Examples Considered	Test Examples Considered	Classes
Reuters – RCV1	8,06,791 documents	6,01,255	2,02,105	4
WebKB	8500 web pages	3200	1250	4
20 newsgroups	19,974 Netnews documents	11500	7600	20
Conference	1200 web pages	870	290	2
Yahoo Web page categories	10,500 pages	8500	2200	5

Reuters-RCV1: Reuters [47] is the collection of manually categorized dataset by Reuters Ltd. which consists of 8 million English news stories. The model classifies Reuters-RCV1 documents into range of international topics, including politics, business, sports, and science.

WebKB dataset: WebKB [52] dataset is a collection of web pages from CMU group World Wide Knowledge Base (Web->KB) project, these 8500 web pages are gathered from different universities computer science departments being manually labelled into 4 classes being, student (1641 pages), faculty (1124 pages), course (930 pages) and other (3764 web pages).

20 newsgroups: 20 newsgroups [45] dataset is an assemblage of nearly 20,000 documents that were gathered from 20 different Netnews newsgroups classified into different categories: alt.atheism, comp.graphics, comp.os.ms-windows.misc, comp.sys.ibm.pc.hardware, comp.sys.mac.hardware, comp.windows.x, misc.forsale, rec.autos, rec.motorcycles, rec.sport.baseball, rec.sport.hockey, sci.crypt, sci.electronics, sci.med, sci.space, soc.religion.christian, talk.politics.guns, talk.politics.mideast, talk.politics.misc, talk.religion.misc

DBLP Conference dataset: 1200 web pages related to computer science conference homepages are crawled from DBLP [49] website to be classified into positive and negative classes.

Yahoo Web page categories: Yahoo Web [51] directory created for highlighting crawler-based listings under Yahoo search, thousands of web pages of crawler based listings are organized into 5 classes namely acquisitions, advertising, litigation, services and news.

4.2 Transformation for Datasets

The data is transformed into native tensorflow framework format, i.e. TFRecord for optimizing the input pipeline. Training and test data is stored as a sequence of binary strings in TFRecord file. To fetch data efficiently we serialize our input data and store it in a set of TFRecord files.

Different tensorflow list files are used like tf.train.Int64List, tf.train. FloatList and tf.train.BytesList to store the transformed crawled data in native format.

4.3 Data Input Pipeline Optimization

The transformed dataset is composed of:

1. Training dataset consist of 27 TFRecord files, each of size ∼95MB, where each TFRecord file contains 5500 target web pages in serialized binary format hosted on laptop machine to be classified into different categories.
2. Test/validation dataset consist of 7 TFRecord files, each of size ∼95MB, where each TFRecord file contains 5500 validation set web pages for evaluating test data effectiveness.

For optimizing the input pipeline various parameters was evaluated using following configuration:

1. Data loading and pre-processing is performed by using multithreading using Autotune TFRecordDataset arguments which are related to num_parallel_calls when calling interleave () and map () function for dataset transformation.
2. Multiple CPU cores are used to pre-fetch the next batch of input data as soon as TPU processes the current training step, this way the TPU will be utilized efficiently apart from the transfer time taken by data from the CPU to the TPU, and this way model training steps will run much faster and efficient.

4.4 Evaluation Metrics

Evaluation metrics namely macro averaged precision, macro averaged recall, macro averaged F1 score, Mathew's correlation coefficient and accuracy are considered for the evaluation of the model using cross-validation or test data set for web page classification. These evaluation metrics are calculated from all the classes.

Table 7 Confusion matrix

True/Actual			
Positive	Negative		
TP	TN	Positive	Predicted
FP	FN	Negative	

For multi-class web page classification problem, we compute precision and recall for each class label and average the values to get the overall macro average precision and macro average recall.

Based on Confusion matrix as in Table 7 definition of TP, TN, FP, FN is:

- Actual positive web pages that are correctly classified positives are called *true positives* (TP);
- Actual positive web pages that are wrongly classified negatives are called *false negatives* (FN);
- Actual negative web pages that are correctly classified negatives are called *true negatives* (TN);
- Actual negative web pages that are wrongly classified positives are called *false positives* (FP).

1. Precision: It identifies how many times true positive web pages are predicted out of total positive web pages.

$$\boldsymbol{Precision} = \frac{\boldsymbol{TP}}{\boldsymbol{TP + FP}} \times 100$$

2. Recall: It identifies how many times true positive web pages are predicted out of total correctly predicted web pages.

$$\boldsymbol{Recall} = \frac{\boldsymbol{TP}}{\boldsymbol{TP + FN}} \times 100$$

3. F1-score: The F1 score is the harmonic mean of the precision and recall.

$$\boldsymbol{F1 - score} = \frac{\boldsymbol{2.Precision.Recall}}{\boldsymbol{Precision + Recall}} \times 100$$

4. Mathew's Correlation Coefficient (MCC): To conquer the unbalanced class issue in dataset another evaluation metrics namely Matthews's correlation coefficient is used for calculating the *correlation coefficient* between

actual and predicted values as:

$$MCC = \frac{TP.TN - FP.FN}{\sqrt{(TP + FP).(TP + FN).(TN + FP).(TN + FN)}}$$

5. Accuracy: Accuracy represents the ratio between the correctly predicted web page instance and all the web page instances in the web page classification dataset:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

4.5 Baseline Comparison

Results are evaluated with three different perspective; firstly the evaluation are figured out without using pre-trained models and metrics are evaluated using traditional ML baseline approaches, afterwards the transfer learning based pre-trained models are used to calculate the results and compared with proposed model evaluation statistics, and then the effectiveness of knowledge fusion is evaluated by incorporating knowledge graph embeddings into pre-trained model along with deep inception on top of it.

Traditional ML Approaches

Traditional ML algorithms like k-nearest neighbors (KNN), Support Vector Machine (SVM) and deep learning algorithms like Artificial Neural Network's (ANN) are used to excerpt the distinguish feature sets and train the model using these unique features.

Pre-Trained Model Approaches

Transfer learning based approaches like transformer encoder based pre-trained bidirectional BERT with its default Base model configuration is computed on input token sequence to classify web pages into pre-defined categories.

Knowledge Fusion Based Approach

Incorporating structured knowledge from task specific domain knowledge graph can enhance pre-trained word embeddings as in BERT in order

to amalgamate extra factual representations to facilitate the web page classification process.

4.6 Results for the Proposed Methodology

Different evaluation metrics as summarized in Table 8 are computed on training and test examples using k-nearest neighbours (KNN), Support Vector Machine (SVM), Deep Neural Networks, Pre-trained default BERT, and BERT with fused Knowledge Embeddings along with Deep Inception approaches on different datasets evaluated.

The model is evaluated from three different perspectives; firstly we compute the evaluation statistics without using pre-trained embeddings and use conventional ML techniques like K-NN, SVM and ANN to extract the feature sets and train the model. We used these conventional approaches as baseline for proposed model evaluation. Secondly we applied pre-trained embedding technique using default BERT Base model. Finally we use an ensemble approach to pre-train the model using BERT fused with knowledge embeddings and use it to fine-tune the labelled dataset using deep inception to compute the desired metrics for evaluation.

Experimental results as in Table 8 shows that using BERT fused with knowledge meanings along with deep inception outperforms benchmark baseline on all evaluation parameters.

Performance of Proposed Approach

The complication we faced while evaluating with transfer learning approaches like BERT Base other deep convolutional neural networks is that as the depth or level of the network is more deepen, the number of features maps increased dramatically. This dilemma will result in a sudden increase in the number of computations required and in terms of parameters to be learned during forward pass. Additionally the use of large filter sizes i.e. 5×5 and 7×7 in previous approaches will result in loss of information as the input matrix dimensions are reduced by larger margin.

The Knowledge fused approach as illustrated in figure 6 uses 1×1 convolution in front of 3×3 and 5×5 filters in different inception blocks to reduce the computational complexity. Use of 1×1 convolutions in the approach will safeguard the spatial dimensions of the input matrix which helps in avoiding vanishing gradient problem when updating the parameters during back propagation along with learning the transitional features in the deeper

Table 8 Evaluation statistics on different datasets

Approach	Datasets	Macro Averaging Precision	Macro Averaging Recall	Macro Averaging F1	Mathews Correlation Coefficient	Accuracy
k-nearest	Reuters -RCV1	0.512	0.535	0.323	0.73	0.79
neighbors	WebKB	0.214	0.223	0.115	0.61	0.70
(KNN)	20 newsgroup	0.605	0.524	0.514	0.63	0.73
	Conference	0.092	0.026	0.011	0.59	0.67
	Yahoo categories	0.324	0.303	0.212	0.53	0.65
Support	Reuters-RCV1	0.563	0.612	0.358	0.75	0.81
Vector	WebKB	0.241	0.213	0.116	0.67	0.71
Machine	20 newsgroup	0.627	0.521	0.512	0.63	0.72
(SVM)	Conference	0.094	0.023	0.011	0.51	0.65
	Yahoo categories	0.312	0.304	0.216	0.58	0.68
Artificial	Reuters-RCV1	0.566	0.617	0.358	0.76	0.81
Neural	WebKB	0.294	0.217	0.127	0.61	0.75
Network	20 newsgroup	0.601	0.522	0.516	0.75	0.81
(ANN)	Conference	0.095	0.025	0.010	0.58	0.63
	Yahoo categories	0.316	0.301	0.211	0.52	0.66
BERT default	Reuters-RCV1	0.618	0.712	0.412	0.79	0.84
	WebKB	0.305	0.216	0.116	0.66	0.73
	20 newsgroup	0.604	0.521	0.513	0.72	0.75
	Conference	0.097	0.022	0.026	0.56	0.62
	Yahoo categories	0.311	0.303	0.219	0.52	0.61
BERT fused	Reuters-RCV1	**0.765**	**0.856**	**0.555**	**0.88**	**0.91**
knowledge	WebKB	**0.411**	**0.247**	**0.163**	**0.79**	**0.87**
embeddings	20 newsgroup	**0.742**	**0.581**	**0.571**	**0.81**	**0.90**
with Deep	Conference	**0.121**	**0.055**	**0.049**	**0.67**	**0.79**
Inception	Yahoo categories	**0.376**	**0.383**	**0.262**	**0.69**	**0.78**

network to have the discriminative features for the web page classification task. Performance of the conventional and knowledge fused BERT model approach using metrics like examples processed per second and test error rates are evaluated as in Table 9.

Table 9 Performance metrics evaluation

	Error Rate	Examples/sec
KNN	0.44	21.2
SVM	0.41	23.5
BERT default	0.36	31.5
BERT fused knowledge embeddings with Deep Inception	**0.29**	**39.7**

Results Hypothetical Analysis

To analyze and evaluate the results obtained for Web page classification task, we model the relationship between different web page hyperlinks and contents of web page and incorporate that into the prediction task. Research hypothesis for proposed model is that BERT representations could be used to create a mechanism for encoding the latent semantics of the web page forming an effective solution for the Web Page Classification problem. This research aims to investigate the effect of using best ensemble technique for classification using pre-trained BERT embeddings instead of training them from scratch using dataset provided for classification.

***Null Hypothesis (H0)*:** The null hypothesis we tested therefore was: there is no performance difference among the ensembles on the dataset. The standard precision (P), recall (R), and F measure metrics were used for comparison which is interpreted using p-value.

H0: p = No performance difference using ensemble technique

***Alternate Hypothesis (H1)*:** The alternate hypothesis of the research states: H_1: If pre-trained language model BERT is employed instead of convolutional neural network built on word2vec embeddings to predict classes to which web page belongs using their textual contents and other Meta tags then the prediction F1-score, accuracy increases.

H1: p $\neq$ No performance difference using ensemble technique

4.7 Statistical Inference

To estimate how accurately proposed model will perform; a cross-validation technique is used to statistically analyze and validate the model performance on the web page classification dataset. Cross-validation, also known as out-of-sample testing is used to test the model ability for prediction on test data

that was not used during the estimation, to get an observation on how the model will generalize on an independent data set.

Cross-validation, a non-parametric method of statistical inference, is used for extracting repeated pieces of data samples from the actual data set. It produced a novel sampling distribution using experimental methods instead of analytical methods for generating specific sampling distribution for the web page classification datasets.

K-fold Cross-validation

K-fold cross-validation, a non-exhaustive cross validation method, is executed by dividing the training data into K folds. During cross-validation execution, the K – 1 fold is considered as training set and the rest made out fold is used as test set. Up to K times, the process is repeated and then the average of K scores is accepted as performance estimation.

To achieve a finer perception of the learning system and more significantly to evaluate the learning deviation, we carry out the 10-fold cross-validation on the Reuters-RCV1training dataset. Table 10 epitomize the outcome of the specific folds throughout the cross-validation for the approaches used. We used cross-validation to validate the performance in terms of model loss and accuracy.

Table 10 Statistical Inference using Cross-Validation technique

KNN	fold 1	fold 2	fold 3	fold 4	fold 5	fold 6	fold 7	fold 8	fold 9	fold 10	Avg.
Loss	0.089	0.086	0.087	0.089	0.088	0.085	0.085	0.082	0.083	0.087	0.085
Accuracy	0.791	0.793	0.792	0.795	0.797	0.793	0.795	0.795	0.793	0.796	0.791
SVM	fold 1	fold 2	fold 3	fold 4	fold 5	fold 6	fold 7	fold 8	fold 9	fold 10	Avg.
Loss	0.078	0.076	0.076	0.077	0.078	0.078	0.077	0.072	0.074	0.077	0.076
Accuracy	0.814	0.817	0.811	0.813	0.816	0.813	0.815	0.814	0.813	0.813	0.814
ANN	fold 1	fold 2	fold 3	fold 4	fold 5	fold 6	fold 7	fold 8	fold 9	fold 10	Avg.
Loss	0.073	0.072	0.071	0.070	0.073	0.071	0.072	0.072	0.073	0.071	0.072
Accuracy	0.811	0.814	0.818	0.815	0.817	0.815	0.814	0.815	0.813	0.816	0.815
BERT default	fold 1	fold 2	fold 3	fold 4	fold 5	fold 6	fold 7	fold 8	fold 9	fold 10	Avg.
Loss	0.069	0.064	0.063	0.067	0.066	0.065	0.068	0.067	0.061	0.066	0.065
Accuracy	0.843	0.841	0.846	0.842	0.847	0.841	0.843	0.844	0.846	0.841	0.844
Proposed model	fold 1	fold 2	fold 3	fold 4	fold 5	fold 6	fold 7	fold 8	fold 9	fold 10	Avg.
Loss	0.049	0.046	0.047	0.049	0.048	0.045	0.045	0.042	0.043	0.047	0.045
Accuracy	0.911	0.913	0.912	0.915	0.917	0.913	0.915	0.915	0.913	0.916	0.914

The cross-validation method was valuable in evaluating the learning deviation in proposed model. As we can inspect from Table 10, that the entire deviation of the learning is moderate, and the proposed knowledge based model exhibits better performance compared to conventional approaches.

In proposed BERT fused KBDI model, cross-validation process is repeated K times, with each of the K subsamples used exactly once as the validation data. The K results are the averaged to achieve a single estimation. We used cross-validation method, as all observations are used for both training and validation, and each observation is used for validation exactly once, which rules out the improvement from independent trials of the algorithms.

Statistical Inference from Table 10 reveals that proposed model exhibits better accuracy during different K fold and the average of K folds of the experimental evaluation, as compared to other algorithms.

5 Conclusion and Future Direction

For the first time as far we know we evaluate the effectiveness of fusing task specific domain knowledge from knowledge graphs by incorporating it into pre-trained bidirectional transformer encoder BERT model along with Deep Inception module for the purpose of classifying web pages into different predefined categories. The model is built using ensemble of pre-trained BERT with structured Knowledge Graph Embeddings to extract features utilizing local information for input tokens in hand and applying deep inception on top of knowledge fused BERT.

The model architecture is optimized using efficient input pipeline for the dataset available with different hyper-parameter values fine-tuned using grid search. The experimental evaluation shows that proposed ensemble with BERT default base configuration and fusion of knowledge embeddings generated from the semantic graph structured knowledge base with deep structured inception blocks outperforms benchmark baselines and provides momentous improvement compared to other transfer learning perspectives for web page classification task. The accuracy resulting from ensemble knowledge embeddings fused with pre-trained BERT having deep inception scores well against other standard baseline classification approaches.

Statistical Evaluation on hypothesis and performance evaluation exhibits that proposed model is more effective than conventional state-of-art approaches used for Web page classification. For the first time, in proposed model we used 1×1 convolutions in different inception blocks to safeguard the spatial dimensions of the input token sequences of web page which helps

in avoiding vanishing gradient problem when updating the parameters during back propagation. To reduce the computational complexity we learns the transitional features as the network goes deeper to have the discriminative features for the web page classification task.

The proposed BERT fused KBDI model exhibits efficient ensemble of state-of-art models using BERT with knowledge graphs to learn long range contextual meanings as well as local relationships between input sequences along with learning deep semantics at multi scale level using deep inception on top of ensemble. The model approach can be replicated to other downstream tasks like fake news detection, sentiment analysis and for recommendation systems. This study will advances the research in classification domain tasks where accuracy for results is more crucial in an optimized manner using fewer learnable parameter overhead and less implication of computational complexity as compared to conventional state-of-art approaches.

Future Work

Dataset DMOZ and other web page classification datasets have single label for individual web page. As web page contents have diverse source of content information related to different categories, it becomes quite challenging for the model to classify web pages in single category with such accuracy and efficiency. Future work may explore multi-label categorization for web page classification to increase success rate. The knowledge fused deep inception approach may be modelled to other downstream task for advancing research in future direction.

Acknowledgements

The authors would like to thank Google Colaboratory for providing free-of-cost TPU for performing our experimentation on efficient web page classification.

References

[1] Brin, S., Page, L. (2012): "Reprint of: The anatomy of a large-scale hypertextual web search engine." Computer Networks. 56(18): 3825–3833. https://doi:10.1016/j.comnet.2012.10.007.

[2] Altingövde, I. S., Özel, S. A., lusoy, Ö., Özsoyoglu, G., Özsoyoglu, Z. M. (2001). Topic-centric querying of Web information resources. Lecture Notes in Computer Science, 2113, 699–711.
[3] De Bra, P. M. E., & Post, R. D. J. (1994). Information retrieval in the World Wide Web: Making client-based searching feasible. Computer Networks and ISDN Systems, 27(2), 183–192.
[4] Menczer, F., Pant, G., & Srinivasan, P. (2004). Topical Web crawlers: Evaluating adaptive algorithms. ACM Transactions on Internet Technology, 4(4), 378–419.
[5] Qi, X., & Davison, B. D. (2009). Web page classification: Features and algorithms. ACM Computing Surveys, 41(2) (article 12).
[6] Chen, R. C., & Hsieh, C. H. (2006). Web page classification based on a support vector machine using a weighted vote schema. Expert Systems with Applications, 31, 427–435.
[7] Selamat, A., & Omatu, S. (2004). Web page feature selection and classification using neural networks. Information Sciences, 158, 69–88.
[8] Chen, R. C., & Hsieh, C. H. (2006). Web page classification based on a support vector machine using a weighted vote schema. Expert Systems with Applications, 31, 427–435.
[9] Ozel, S. A. (2011). A web page classification system based on a genetic algorithm using tagged-terms as features. Expert Systems with Applications, 38(4), 3407–3415.
[10] Kwon, O., & Lee. J. (2000). Web page classification based on k-nearest neighbour approach. IRAL '00: Proceedings of the fifth international workshop on Information retrieval with Asian languages (pp. 9–15).
[11] Selamat, A., & Omatu, S. (2004). Web page feature selection and classification using neural networks. Information Sciences, 158, 69–88.
[12] Sara Meshkizadeh, Amir Masoud Rahmani, Mashallah Abassi Dezfuli (2010), "Web Page Classification based on RL features and Features of Sibling Pages", JCS S, Vol. 8, No. 2.
[13] Nicholas Holden and Alex A. Freitas, (2004), "Web Page Classification with an Ant Colony Algorithm", Parallel Problem Solving from Nature, LNCS, Springer, Vol. 3242, (pp. 1092–1102).
[14] Rung-Ching Chen, Chung-Hsun Hsieh (2006), "Web Page Classification based on a support Vector Machine using a weighted vote schema", Expert Systems with Applications, Vol. 31, Issue 2, (pp. 427–435).
[15] Ribeiro, A., Fresno, V., Garcia-Alegre, M. C., & Guinea, D. (2003). Web page classification: A soft computing approach. Lecture Notes in Artificial Intelligence, 2663, 103–112.

[16] Bollacker, K., et al. (2008): "Freebase: a collaboratively created graph database for structuring human knowledge," in Proceedings of the ACM SIGMOD international conference on Management of data, pp. 1247–1250.
[17] Mitchell, T., et al. (2018): Never-ending learning. Communications of the ACM, 61(5):103–115, 2018.
[18] Erxleben, F., et al. (2014): Introducing wikidata to the linked data web. In Proceedings of the 13th International Semantic Web Conference.
[19] Zhang, X., Zhao, J., LeCun, Y. (2015): Character-level convolutional networks for text classification. In: Advances in Neural Information Processing Systems.
[20] Tai, K.S., Socher, R., Manning, C.D. (2015): Improved semantic representations from tree-structured long short-term memory networks. arXiv preprint arXiv:1503.00075.
[21] Chung, J., et al. (2014): Empirical evaluation of gated recurrent neural networks on sequence modeling. arXiv preprint arXiv:1412.3555.
[22] Conneau, A., et al. (2016): Very deep convolutional networks for text classification. arXiv preprint arXiv:1606.01781.
[23] Kim, Y. (2014): Convolutional neural networks for sentence classification. arXiv preprint arXiv:1408.5882.
[24] Huang, M., Qian, Q., Zhu, X. (2017): Encoding syntactic knowledge in neural networks for sentiment classification. ACM Trans. Inf. Syst. (TOIS) 35(3), 26.
[25] Ozel, S. A. (2011). A genetic algorithm based optimal feature selection for web page classification. In Innovations in Intelligent Systems and Applications (INISTA), 2011 International Symposium on IEEE. (pp. 282–286).
[26] Wang, B. (2018): Disconnected recurrent neural networks for text categorization. In: Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, Long Papers, vol. 1.
[27] Kalchbrenner, N., Grefenstette, E., Blunsom, P. (2014): A convolutional neural network for modelling sentences. arXiv preprint arXiv:1404.2188.
[28] Zhou, C., et al. (2015): A C-LSTM neural network for text classification. arXiv preprint arXiv:1511.08630.
[29] Yang, Z., et al. (2016): Hierarchical attention networks for document classification. In: Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies.

[30] Xiao, Y., Cho, K. (2016): Efficient character-level document classification by combining convolution and recurrent layers. arXiv preprint arXiv:1602.00367.
[31] He, K., Zhang, X., Ren, S., Sun, J. (2016): Deep residual learning for image recognition. In: CVPR, (pp. 770–778).
[32] Szegedy, C., Liu, W., Jia, Y., Sermanet, P. (2015): Going deeper with convolutions. In: CVPR, (pp. 1–9).
[33] Szegedy, C., Ioffe, S., Vanhoucke, V., Alemi, A. (2016): Inception-v4, inception-ResNet and the impact of residual connections on learning. arXiv e-print arXiv:1602.07261.
[34] Peters, M. E., et al. (2018): Deep contextualized word representations. arXiv preprint arXiv:1802.05365.
[35] Radford, A., et al. (2018): Improving language understanding by generative pre-training. https://s3-us-west-2.amazonaws.com/openai-assets/research-covers/languageunsupervised/languageunderstandingpaper.pdf
[36] Howard, J., Ruder, S. (2018): Universal language model fine-tuning for text classification. arXiv preprint arXiv:1801.06146.
[37] Devlin, J., et al. (2018): Bert: pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805.
[38] Yang, B., et al. (2015): Embedding entities and relations for learning and inference in knowledge bases. In International Conference on Learning Representations (ICLR).
[39] Google, https://www.google.com/intl/bn/insidesearch/features/search/knowledge.html, (2014).
[40] Wikidata, http://wikidata.org/, (2012).
[41] Biega, J., et al. (2013): Inside YAGO2s: A transparent information extraction architecture. In Proceedings of the 22nd International Conference on World Wide Web Companion; pp. 325–328.
[42] Minaee., et al. (2021): Deep Learning Based Text Classification: A Comprehensive Review, ACM Computing Surveys (CSUR), vol. 54 (3), pp. 1–40.
[43] Xiaoyu Luo (2021): Efficient English text classification using selected Machine Learning Techniques, Alexandria Engineering Journal, vol. 60(3), pp. 3401–3409.
[44] Bizer, C., et al. (2009): DBpedia-A crystallization point for the Web of data. J. Web Semant; pp. 154–165.
[45] https://kdd.ics.uci.edu/databases/20newsgroups/20newsgroups.data.html

[46] Chai., et al. (2020): Description Based Text Classification with Reinforcement Learning, Proceedings of the 37th International Conference on Machine Learning.
[47] Lewis., et al. (2004): RCV1: A new benchmark collection for text categorization research, The Journal of Machine Learning Research, vol. 5, pp. 361–397.
[48] Li., et al. (2021): Word embedding and text classification based on deep learning methods, MATEC Web of Conferences, https://doi.org/10.1051/matecconf/202133606022
[49] Melanie., et al (2005): A Duplicate Detection Benchmark for XML (and Relational) Data.
[50] Jiang., et al. (2017): Integrating Bidirectional LSTM with Inception for Text Classification, 2017 4th IAPR Asian Conference on Pattern Recognition (ACPR).
[51] https://en.wikipedia.org/wiki/Category:Yahoo!
[52] http://www.cs.cmu.edu/afs/cs.cmu.edu/project/theo-20/www/data/webkb-data.gtar.gz
[53] Pouyanfar, et al. (2017): An efficient Deep Residual-Inception Network for Multimedia Classification, 2017 IEEE International Conference on Multimedia and Expo (ICME), https://doi.org/10.1109/ICME.2017.8019447

Biographies

Amit Gupta is currently a Research Scholar with the Department of Computer Science and Engineering, Punjab Engineering College (Deemed to be University), Chandigarh, India. He received the M.E. degree in Software Engineering from Thapar University, Patiala, and the B.Tech. degree in Information Technology from Kurukshetra University, Kurukshetra. He is working

as Scientist C in National Informatics Centre, Ministry of Electronics and Information Technology, Government of India, and has more than 11 years of experience in delivery of government IT services and initiatives of Digital India. His research interests include Deep Learning, Internet of Things, Virtual and Augmented Reality, and the Artificial Intelligence.

Rajesh Bhatia is currently working as a Professor in the Department of Computer Science and Engineering at Punjab Engineering College (Deemed to be University), Chandigarh, India. He received his Ph.D. and M.E. degrees in Computer Science Engineering from Thapar Institute of Engineering Technology (Deemed to be University), Patiala, India. He has received B. Tech. degree from Dr. B. Ambedkar Marathwada University, Aurangabad, India. He has more than 25 years of Teaching and Research experience. His research areas include Automated Software Debugging, Semantic Software Clones detection and Automated Test Cases Generation, Information Retrieval, and Search Based Software Engineering. He is also undertaking various Sponsored Research Projects. He has about 85 research publications in various reputed journals and conferences. He is member of various national and international professional bodies such as CSI, ISTE, IEEE, ACM-SIGSE & ACEEE. He has been conferred with National Science and Technology Database award on National Science Day-2021.