IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 26 June 2022, accepted 10 August 2022, date of publication 25 August 2022, date of current version 6 September 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3201542

RESEARCH ARTICLE

K-LM: Knowledge Augmenting in Language Models Within the Scholarly Domain

VIVEK KUMAR[1], DIEGO REFORGIATO RECUPERO[1], RIM HELAOUI[2], AND DANIELE RIBONI[1]

[1]Department of Mathematics and Computer Science, University of Cagliari, 09124 Cagliari, Italy
[2]Philips Research, High Tech Campus, 5656 AE Eindhoven, The Netherlands

Corresponding author: Vivek Kumar (vivek.kumar@unica.it)

This work was supported by the European Unions's Horizon 2020 Marie Sklodowska-Curie Industrial Training Network Program—PhilHumans under Agreement 812882 (https://www.philhumans.eu/).

ABSTRACT The use of superior algorithms and complex architectures in language models have successfully imparted human-like abilities to machines for specific tasks. But two significant constraints, the available training data size and the understanding of domain-specific context, hamper the pre-trained language models from optimal and reliable performance. A potential solution to tackle these limitations is to equip the language models with domain knowledge. While the commonly adopted techniques use Knowledge Graphs Embeddings (KGEs) to inject domain knowledge, we provide a Knowledge Language Model (K-LM) to use the Resource Description Framework (RDF) triples directly, extracted from world knowledge bases. The proposed model works in conjunction with Generative Pretrained Transformer (GPT-2) and Bidirectional Encoder Representations from Transformers (BERT) and uses a well-defined pipeline to select, categorize, and filter the RDF triples. In addition, we introduce heuristic methods to inject domain-specific knowledge in K-LM, leveraging knowledge graphs (KGs). We tested our approaches on the classification task within the scholarly domain using two KGs, and our results show that our proposed language model has significantly outperformed the baselines and BERT for each KG. Our experimental findings also help us conclude the importance of relevance of KG used over the quantity of injected RDF triples. Also, each of our proposed methods for injecting the RDF triples has increased the overall model's accuracy, demonstrating that K-LM is a potential choice for domain adaptation to solve knowledge-driven problems.

INDEX TERMS Deep learning, machine learning, knowledge graphs, knowledge graphs embeddings, GPT-2, BERT.

I. INTRODUCTION

Humans have the innate ability to elicit previously acquired knowledge and integrate it with newly learned concepts to solve tasks at hand. However, we have made remarkable progress in developing intelligent systems to mimic human-like abilities in recent times, still, these systems are limited in scalability and are very task-specific. For instance, the transfer learning [1] approach has left a mark in cross-domain adaptation, but it is limited to interrelated domains application. Similarly, the general language model (LM) Bidirectional Encoder Representations from Transformers (BERT) [2], pre-trained on Wikipedia[1] and BookCorpus,[2] has given promising results on specific Natural Language Processing (NLP) downstream tasks. But in the case of cross-domain adaptability, it lacks task-specific and domain-related knowledge, and hence more detailed fine-tuning strategy analyses are necessary to further improve the performance [3]. To overcome the constraint of domain-adaption, approaches based on augmenting additional knowledge to LMs have turned out to be effective. One way to

The associate editor coordinating the review of this manuscript and approving it for publication was Rongbo Zhu.

[1]https://www.wikipedia.org/
[2]https://huggingface.co/datasets/bookcorpus

inject general world knowledge is represented by Knowledge Graphs (KGs), which are well-known interlinked structured knowledge, comprised of triples [4]. A semantic triple, or RDF triple or triple, is the atomic data entity in the RDF data model. As the name indicates, a triple is a set of three items, *subject (s), predicate (p)*, and *object (o)*, that encodes the semantic data (e.g., <Deep Learning, subdomain, Machine Learning>). The triples are represented as (s, p, o), where the predicate (p) is the relationship between the subject and the object. A few of the popular general knowledge bases available in the public domain include WordNet[3] [5], Cyc[4] [6], DBpedia[5] [7], YAGO[6] [8], Freebase[7] [9], NELL[8] [10], and Wikidata[9] [11]. The existing methods for injecting the world knowledge into the classification models use KGs in the form of Knowledge Graphs Embedding (KGE). KGEs are low-dimensional representation of knowledge graph's entities and relations while preserving their semantic meaning [12] and are useful for tasks such as KG completion [13], relation extraction [14], [15], entity classification [16], [17], entity resolution [16], [18], etc. In the existing literature a plethora of knowledge representation approaches such as TransE [19], TransH [13], TransR [20], DistMult [21], ComplEx [22], RotatE [23], HolE [24], ConvE [25], ConvKB [26], DKRL [27] are available to transform triples into KGE. KGE has proven to be useful in incorporating world knowledge, but it is still debatable if the KGEs sufficiently capture KGs semantics [28]. Analyses from this work indicate that leveraging KGEs for semantic interpretability (as in the case of word embeddings) may seem intuitive, but this is not always the case because the performance of KGE is limited and is heavily dependent on the characteristics of the dataset. This conclusion is based on the evaluation of the semantic representation of KGEs. It is observed that for given KG entities, KGEs can learn certain semantic features, but this learning is non-uniform due to the varying quality of semantic representation across different entities within the dataset. These findings raise questions about the applicability and efficacy of KGEs not only for semantic capturing but also for link prediction [29], [30], [31] and triple completion.

Two of the available existing works tackled the challenge of semantics capturing associated with KGE, by using RDF triples as a direct source of knowledge [32], [33]. However, these works are far from mature and lack to answer some crucial knowledge fronts. For instance, they do not discuss the quantification of knowledge injection for achieving optimal classification performances. They also do not discuss the impact of using general KGs in domain-specific tasks. These limitations motivated our investigation of using triples as the source of semantic knowledge, and the challenges that arise in domain-specific and open-domain NLP downstream tasks. In this paper, we propose a LM (K-LM), which is the combination of BERT [34] and Generative Pretrained Transformer (GPT-2) [35], [36] for text classification. K in K-LM stands for knowledge, and LM is the language model. K-LM leverages world knowledge by incorporating additional knowledge in the form of triples to mitigate the prevalent knowledge gap scenarios. We have performed experiments on the *scholarly* domain using KGs of the same domain. Furthermore, we also introduce `Deterministic` and `Non-deterministic` strategies to seed the triples in the proposed LM for achieving optimal results in domain-specific and open-domain classification tasks. The contributions of our paper are, therefore:

- We provide an LM to directly inject world knowledge in the form of triples to address the knowledge gaps in domain adaptation, i.e., scholarly domain in this research work.
- We introduce different heuristic methods to select and filter the triples in the LM and provide a detailed analysis of the observed experimental results.
- We employ Deterministic and Non-deterministic strategies for the semantic evaluation of the input triples concerning the context-dependency.
- We experimentally compare the impact of quantity (varying number of triples injected per token in the input texts and size of KGs) vs. relevance (semantic co-relation of KGs for the given context) in the knowledge injection process.
- Finally, we provide a comprehensive discussion of the performances of different KGs used, their impact on knowledge injection, and the importance of the context captured by the triples in the process.

The rest of the paper is organized as follows. Section II presents some literature background and related works. Section III describes the motivations behind this work, provides details of the dataset, preprocessing steps, and KGs used for the experiments. It also includes the necessary notations used throughout the paper. Section IV presents the structure of the K-LM model and discusses the techniques and concepts developed to feed the triples in the K-LM. Section V presents the experimental evaluation and in-depth analyses of the obtained results. Finally, Section VI concludes the paper and presents the directions for future research.

II. RELATED WORK

This section briefly reviews the existing state-of-the-art techniques for augmenting knowledge in the form of triples for domain-specific and open-domain tasks. The pre-trained BERT model trained on cross-domain text corpora such as BookCorpus[10] and Wikipedia[11] has achieved significant performance on several NLP downstream tasks. Despite this

[3] https://wordnet.princeton.edu/
[4] https://cyc.com/
[5] http://downloads.dbpedia.org/wiki-archive/
[6] https://yago-knowledge.org/
[7] https://developers.google.com/freebase/
[8] http://rtw.ml.cmu.edu/rtw/
[9] https://www.wikidata.org/wiki/Wikidata:Main_Page
[10] https://huggingface.co/datasets/bookcorpus
[11] https://en.wikipedia.org/wiki/Main_Page

improvement, the transformer model lacks task-specific and domain-related knowledge, which can contribute to further improvements. To address the limitations of domain knowledge gaps in cross-domain knowledge-driven NLP tasks, the work proposed a BERT-based text classification model called BERT4TC [37]. The proposed model focuses on constructing auxiliary sentences and converting the original classification task into a sentence pair with a binary set of new categorical labels. Another work proposed a knowledge-enabled language representation model of BERT (K-BERT) with KGs [32]. The triples were used in the sentences as domain knowledge along with soft-position and a visible matrix to limit the impact of knowledge to tackle the domain. The purpose of the visible matrix was to control the flow of a huge amount of knowledge (also called Knowledge Noise (KN)), as it may lead to a change in the sentence from its actual meaning. Structurally, the English language operates on tokens or word-level embeddings, while the model proposed in this work used character-level embeddings; therefore, this constraint limits its application to NLP tasks in Chinese language. Furthermore, a severe limitation of the work was that it did not provide any evaluation method to determine the suitability of triples and order them according to the given context. The work performed in [33] proposed an attention mechanism-based Deep Learning (DL) model that could use knowledge graphs as knowledge support for the task at hand. It can also be considered the first work that attempted to directly incorporate the KGs in the form of triples. It was about a convolution-based model that learns representations of knowledge graph entities and relation clusters by reducing the attention space. The outcome of this work shows that the proposed model is suitable for the classification tasks at hand while being trained on significantly less training data when it has access to world knowledge resources such as KGs.

However, the works mentioned above provide methods to use world knowledge, but they significantly lack on several knowledge fronts, such as novel methods to select best-fit triples based on context, quantification of triples injection, and challenges of KN associated with knowledge injection. To bridge these knowledge gaps, our approach introduces a robust pipeline to select and inject triples, starting from the input sentences themselves. Our work also takes into account the "context-aware" and "context-unaware" dependencies of the LM and their impact on the knowledge integration process. We implement novel approaches that allow us to custom feed and rank the input triples to address this challenge. Our approaches are comprised of `Non-deterministic` and `Deterministic` methods, which help in the quantification of knowledge injection and generalization of direct integration of triples for optimal knowledge injection in the LM. In the context of generalization, we have used two KGs as a source of domain knowledge and performed experiments with incremental integration of triples to draw conclusions about the relevance of used KGs for the task at hand.

III. PROBLEM FORMULATION, DATASET, AND PREPROCESSING

This section presents the problem statement we are tackling, the details of the used dataset, the preprocessing strategies, and the KGs employed to carry out the experiments.

A. PROBLEM FORMULATION

The main goal of our work is to infuse domain knowledge in pre-trained language models to equip the models with additional knowledge available in the form of KGs and leverage the added knowledge to achieve higher accuracy in the classification tasks. In this work, we have used the K-LM model to tackle a binary classification task. More precisely, for a given text $\boldsymbol{t}$ and a target class $\boldsymbol{c}$, the objective is to infer a function f(t, c) $\rightarrow$ l that computes 1 if $\boldsymbol{t}$ belongs to the class $\boldsymbol{c}$, 0 otherwise. Here $\boldsymbol{l}$ is the binary label that can only take values in {0, 1}. To inject the domain knowledge in K-LM, we have used two KGs, namely AI-KG and AI-KG-Small. The details of K-LM and KGs are provided in subsequent sections. Our work introduces novel methods of using world knowledge and different processes of integrating the domain knowledge, which is fundamentally different from transfer learning. Our study also aims to benchmark the quantity of knowledge injection and introduce novel methods for filtering the triples to inject into classification models.

B. DATASET DESCRIPTION

We have performed our research study on the *Scholarly Domain*. The dataset we have considered is constituted by 30,023 abstracts of research papers in Computer Science from 2001 to 2019. The labels 1 and 0 represent if the underlying research paper belongs to the Artificial Intelligence (AI) domain or not. The distribution of the dataset is: Label-0 = 9,720, Label-1 = 20,303. From the given distribution of the scholarly dataset, it is evident that it is unbalanced, and the majority class is label 1, representing that most of the papers belong to the AI domain. To prepare the textual data, we have adopted standard preprocessing steps such as lowercasing the text, special characters removal, and tokenization [38], [39], [40], [41], [42]. The input dataset is further divided into the train, validation, and test sets. The train set is used to fine-tune the model, while the model's performance evaluation is done on the validation and test sets. The distribution of test, validation and train sets of the dataset is 60%, 20%, and 20%, respectively. For ease of understanding, we have listed the notations used throughout the paper in Table 1.

C. KNOWLEDGE GRAPHS

For our work, we have used KGs as a medium to infuse world knowledge for *Scholarly Domain* and the details of the used KGs are mentioned below.

- **AI-KG**- The Artificial Intelligence Knowledge Graph (AI-KG) is an automatically generated large-scale knowledge graph comprised of 857,658 research entities. AI-KG consists of 1,2 million statements and

TABLE 1. Notations used.

Uni-sub-triple	Triple extracted from AIKG and AI-KG-Small whose subject is a unigram (one word).
Bi-sub-triple	Triple extracted from AIKG and AI-KG-Small whose subject is a bigram (two words).
Tri-sub-triple	Triple extracted from AIKG and AI-KG-Small whose subject is a trigram (three words).
AI-KG	The Artificial Intelligence Knowledge Graph used as source of world knowledge.
AI-KG-Small	The subset of Artificial Intelligence Knowledge Graph.
Full KG	When all the triples of an entire input KG, are used.
K-LM	The Language Model proposed in the paper which is combination of GPT-2 and BERT.
N	The total number of triples present in the given KG (AI-KG and AI-KG-Small in our case).
U	The total number of unique triples present in the given KG. Unique triples are defined as the triples having distinct *'subject'* entity.
L	The length of the entity 'subject' of the triples. It is measured by the number of words present in the subject. For $\{l = 1, 2, 3\}$ a given triple is called *Uni-sub-triple*, *Bi-sub-triple* and *Tri-sub-triple*, respectively.
TIT	For a given KG and dataset used for our experiments, the total injectable triple is the number of triples that can be injected into the input sentence while fine-tuning K-LM. A triple is considered *'injectable'* when its subject is present in the input sentence.

14 million triples which are extracted from 333K research publications belonging to the AI domain. AI-KG describes 5 types of entities namely methods, metrics, materials, tasks and others linked by 27 relations. AI-KG is a rich source of world knowledge and is designed to support various intelligent services for analyzing and making sense of research dynamics, supporting researchers in their day-to-day work, and informing the decision of founding bodies and research policymakers. In this work, we have used AI-KG as a source of domain knowledge to infuse task-specific knowledge in the domain-adaptation scenario. AI-KG is generated by using the automatic pipeline mentioned in [43] and [44] before doing the transitive closure and linking the entities to *Wikidata* and CSO. The AI-KG dump can be downloded in *.ttl* format from https://scholkg.kmi.open.ac.uk/.

- **AI-KG-Small**- It is the reduced version of AI-KG that is generated by post-processing of AI-KG by linking the entities to *Wikidata* and CSO. AI-KG-Small contains 12,094 triples, and the purpose of using AI-KG-Small is to study the "quantity vs. relevance" effect of triples for the given dataset.

IV. KNOWLEDGE-LANGUAGE MODEL

In this section, we provide in detail the concepts related to K-LM, followed by the framework and implementation of K-LM. At last, we describe the techniques developed to feed the triples in the K-LM.

A. CONCEPTS RELATED TO K-LM

This subsection describes the concepts and terminologies associated with K-LM used throughout the rest of the paper.

- **Unique Triples**- For a given KG having N triples, we define U as the triples having a distinct "subject" entity. Inherently, the triples are arranged in key-value pairs where key $\rightarrow s$ and value $\rightarrow \{p, o\}$. Consider the sentence - *"Italy is home of culture and cuisine"*, and a set of triples (Italy, Country, Europe), (Italy, famous, Pizza), (Italy, famous_for, Roman Architecture), and (Amsterdam, Capital_of, Netherlands). In the process of knowledge injection, when token ***Italy*** is queried, the *look-up* table returns all the triples with the "subject" ***Italy***, and they are organized as follows: ('Italy', *['country, Europe', 'famous, Pizza', 'famous_for, Roman Architecture']*). In this example $\{N = 4\}$ & $\{U = 2\}$. Generating a KG aims at holistically covering the scope of the domain. But all the triples contained within the KG are not necessarily useful for a specific dataset. In this context, we propose U as the parameter to measure the relevance of the KG to the dataset. Quantitatively, the less the difference between N and U is, the more relevant the KG is.
- **Max Entity**- It is the parameter that controls the number of branches that can be associated with the tokens in the input sentence while injecting knowledge. For $\{Max\_Entity = 1\}$ one triple is associated with the corresponding token of the input sentence and for $\{Max\_Entity = 2\}$, two triples will be associated with the same token, and so on. Note that even if $\{Max\_Entity \geq 2\}$, the number of triples associated with a particular token solely depends upon the triples' availability in the KG. Should the number of available triples for a particular token be less than the value of *Max_Entity*, only the available triple(s) will be associated. We have demonstrated this trivial case in Figure 1. Consider the input sentence in Figure 1 and the tokens within; ***graph***, and ***embedding***. We assume that only one triple is available for token ***graph*** and more than two triples are available for token ***embedding*** for triples injection. Therefore, for $\{Max\_Entity = 2\}$ the resulting input sentence has only one triple (***graph***, *uses, human scientific creativity*) associated with the token ***graph*** while the token ***embedding*** has two triples associated, i.e., (***embedding***, *represents, vectors in high-dimensions*) and (***embedding***, *are, compressed representation*). However, a sentence tree can have multiple branches, i.e., several values of *Max_Entity*, but its depth is fixed to one, which means the entity names in triples will not further derive branches iteratively. For our experiments, we have used $Max\_Entity \in \{1, 2, 3, 4, 5\}$.

B. ARCHITECTURE OF K-LM

Implementation of K-LM is a two-stage process and is executed by the `K-LM Triple Selection` and `K-LM Classification Modules`. The architecture of K-LM

is presented in Figure 1. The objective of the K-LM Triple Selection Module is to filter and rank (if applicable) the triples provided in AI-KG or AI-KG-Small through a well-defined pipeline. The objective of the second module is to use the triples selected through the K-LM Triples Selection Module for injecting knowledge into the input sentences and then perform the classification task on the provided dataset. The K-LM Classification Module of K-LM takes inspiration from the LM model proposed in [32], which was used for binary classification tasks, such as the classification of online reviews for books, hotels, and shopping in the Chinese language.

As mentioned before, K-LM is the combination of GPT-2 and BERT. The GPT-2 is used in the K-LM Triples Selection Module for ranking the triples, while BERT is used in the K-LM Classification Module. Our central idea is to use the available world knowledge, independent of a particular experimental dataset; therefore, GPT-2 is not used for generating triples for our work because that will lead to knowledge injection based on the used dataset.

1) **K-LM Triples Selection Module-** This module performs the selection, filtering, and ranking of the triples, which are further used in the K-LM Classification module, for knowledge injection. The pipeline of the K-LM Triple Selection module as shown in Figure 1 is comprised of four sub-modules which are explained below.
   - **Input Triples-** This sub-module is provided with the randomly distributed triples from the input KGs, i.e., AI-KG or AI-KG-Small.
   - **Triples Categorization-** This sub-module performs categorization for generating input triples for the *Triples Injection Methods*. Four distinct categories of triples are produced by this sub-module, namely *Uni-sub-triple*, *Bi-sub-triple*, *Tri-sub-triple*, and *Full KG*.[12] While *Full KG* contains all the triples of the input KG, *Uni-sub-triple*, *Bi-sub-triple*, and *Tri-sub-triple* are categorized based on the L of the input triples.
   - **Triples Injection Methods-** This sub-module implements two methods, *Forward* and *Reverse Injection* for triples selection. The output of this sub-module consists of disordered triples. The *Forward Injection* method employs four techniques: *Uni-sub-triple Injection*, *Bi-sub-triple Injection*, *Tri-sub-triple Injection* and *Full KG Injection*. The *Reverse Injection* method uses the *Reversion Full KG Injection* technique to select the triples. At this stage, the disordered triples have two possibilities:
     a. They are either injected directly into the K-LM Classification Module or
     b. They are sent to the sub-module *Triples Ranking* to generate the ordered triples.

     This scenario leads to the development of two approaches, namely Non-deterministic and Deterministic in the pipeline, representing the uniqueness and added advantage of K-LM. For ease of understanding, the *Forward* and *Reverse Injection* and the Non-deterministic and Deterministic approaches are explained separately in subsection IV-C.
   - **Triples Ranking-** The ranking of triples finds extensive uses when the experimental dataset is small and the input KG is sparsely related to the given dataset. Therefore, this sub-module serves the purpose of infusing the triples based on the context of input sentences for optimal domain knowledge integration. This sub-module takes the disordered triples as input from the *Triples Injection Methods* and outputs the ranked triples. The process of ranking the triples is explained below in detail. The triples within the KGs (AI-KG and AI-KG-small) are stored in the *look-up* table and are fetched to the sub-module *Triples ranking*. To rank the triples, we have used GPT2LMHeadModel,[13] which is the GPT-2 Model transformer with a language modeling head on top, and the GPT-2 Tokenizer.[14] The GPT-2 model has about 1.5 billion parameters trained on a dataset of 8 million web pages, which makes it very useful to predict the most likely word by interpreting a given sequence of words. In our case, for the given set of triples and context, GPT-2 takes as input each triple iteratively and inserts it in the input sentence providing a score for each sentence thus formed.[15] The outputs are the scores for each sentence used with each triple, which we call probability score $\boldsymbol{P}$. The value of $\boldsymbol{P}$ is related to the suitability of the underlying triple in the given context of the sentence. More precisely, for a given input sentence $i_s = \{w_o, w_1, w_2, \ldots, w_n\}$, where $\{w_i\}$ are the words of i_s and the set of triples $t = \{w_0, p_0, o_0\}, \{w_0, p_1, o_1\}, \{w_0, p_2, o_2\}$; the score function is given by $f_s(i_s, t_0^n) \rightarrow \{P_0, P_1, \ldots, P_n\}$, where $P_0, \ldots, P_n$ are the probability scores for each triple $\in t$. Hence, in the above-mentioned set of triples t, $\{w_0\}$ represents the common subject (and is also present in i_s) while $\{p_0, p_1, p_2\}$ and $\{o_0, o_1, o_2\}$ are the respective predicates and objects of the three triples. In this case, the score function f_s thus returns three probability scores (for each of the three triples having a common subject) $\mathbf{P}_0$, $\mathbf{P}_1$, and $\mathbf{P}_2$. Based on the obtained probability scores of the sentence,

[12] Refer to Table 1 for the definition of *Uni-sub-triple*, *Bi-sub-triple*, *Tri-sub-triple* triples and *Full KG*.

[13] https://huggingface.co/transformers/model_doc/gpt2.html##gpt2lmheadmodel

[14] https://huggingface.co/transformers/model_doc/gpt2.html##gpt2tokenizer

[15] https://github.com/huggingface/transformers/issues/1009

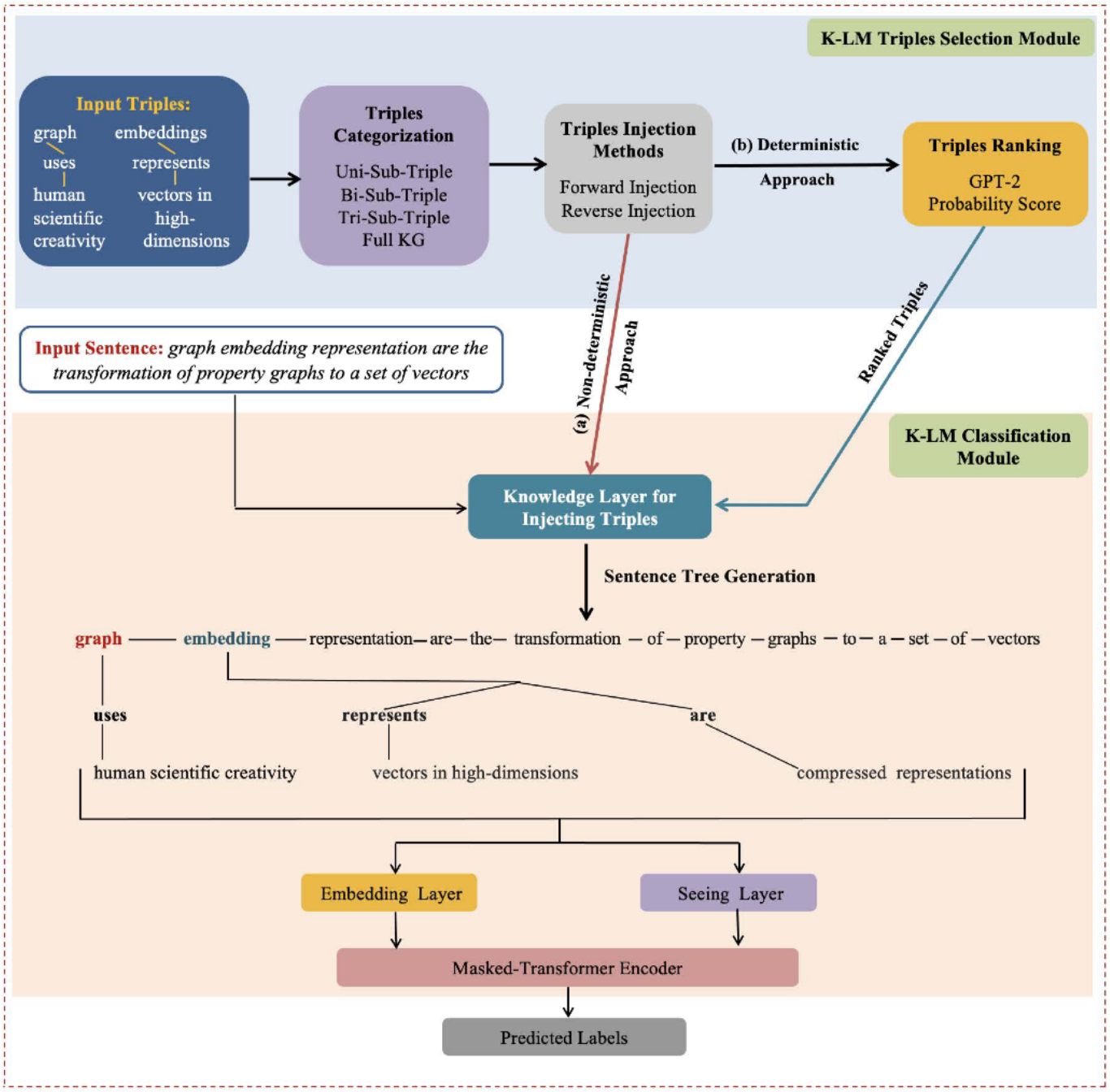

FIGURE 1. K-LM architecture that contains two modules (a) K-LM Triples Selection and (b) K-LM Classification module. Module (a) performs the selection and categorization of the triples and ranks them by employing Non-deterministic and Deterministic approaches. Further, the processed triples are used as the source of domain knowledge in module (b) for the classification task.

the triples are ranked (ordered) in decreasing value of relevance, i.e., the first triple in the ranking is the best fit for the input sentence. The relevance of triples for the input KG is measured by ***TIT***. The higher the value of ***TIT*** is, the more relevant the KG is to the dataset.

2) **K-LM Classification Module**- It consists of four elements, i.e., knowledge layer for injecting triples, embedding layer, seeing layer, and masked-transformer encoder. For an input sentence, the `Knowledge Layer for Injecting Triples` first injects the triples processed from the `K-LM Triples Selection Module`. The triples are injected in the form of (s, p, o). The injection of triples transforms the input sentence into a sentence tree equipped with domain knowledge. The sentence tree thus generated is simultaneously fed to both embedding and the seeing layers. The embedding layer generates the embedding representation of the flattened input sentence tree, while the seeing layer generates a visible matrix.

The visible matrix contains the semantic information of the actual input sentence and provides control over the extent of mutual interaction between the tokens within the input sentence. This mechanism prevents the deviation and false semantic changes from the actual meaning of the original sentence by minimizing the semantic interference and KN caused by triples injection. K-LM is built on top of the BERT (transformer) model; therefore, it uses the same token, position, and segment embeddings approach to generate the embedding representation. The difference between the embedding layer of K-LM and BERT is the input type, i.e., K-LM takes sentence tree as input instead of token sequences. Further, the cumulative outputs of embedding and seeing layers are fed to the masked-transformer encoder (which is a modified BERT model in our case), that executes the binary classification task at hand, i.e., predicts each input data point as "paper belongs to AI-domain" or not.

C. TRIPLES SELECTION TECHNIQUES

The core idea of the `K-LM Triple Selection Module` is to provide the `K-LM Classification Module` with selected triples for knowledge integration. This subsection looks at the `K-LM Triple Selection Module` at the fine-grained level and provides further details about the internal mechanism of its sub-modules; *Triples Injection Methods* and *Triples Ranking*. The selection of triples follows the pipeline presented in Figure 2, where the input triples are first passed to *Triples Categorization*. In this sub-module, the triples are categorized based on the size ***L*** of the triples. The categorized triples are further processed in the sub-module *Triples Injection Methods* following *Forward* and *Reverse Injection*. These two methods uncover the behavior of K-LM in terms of integrating triples and the techniques we have developed for the optimal selection of triples. *Forward* and *Reverse Injection* methods and interdependencies with the other sub-modules are explained below.

1) **Forward Injection**- Inherently K-LM injects the triples based on the "sequence of occurrence" of words/tokens in the input sentence for a given set of input triples. For instance, consider the input sentence *"**graph embedding representation** is the transformation of property of graphs to a vector or a set of vectors"* and the set of triples:
   a) <***graph***, *uses, human scientific creativity*>
   b) <***embedding***, *represents, vector in high-dimensions*>
   c) <***graph embedding***, *uses, semi-supervised learning strategy*>
   d) <***graph embedding representation***, *incorporates, structural and topological feature*>

   In the knowledge injection process, when the K-LM queries the tokens of the input sentence in the *look-up* table, it first takes into account the token ***graph*** followed by ***embedding*** based on their "sequence of occurrence". Since the triples with the subject ***graph*** and ***embedding*** are already available as input triples, the K-LM automatically integrates these two triples in the sentence. The injection of these triples does not leave any further possibility to inject the triples with subjects < ***graph embedding*** > or < ***graph embedding representation*** >. We call this the default behavior of K-LM in which the priority of triples decreases through *Uni-sub-triple*, *Bi-sub-triple* and *Tri-sub-triple* triples. In simple words, the triples having the subject comprised of unigram get the highest priority. The advantage of this method is that it ensures the injection of the highest number of *Uni-sub-triple* triples because the existing number of *Uni-sub-triple* triples in AI-KG and AI-KG-Small is far more than *Bi-sub-triple* and *Tri-sub-triple* triples. The injection of the highest number of *Uni-sub-triple* triples is not always the optimal solution for incorporating domain knowledge, and it is obvious that the *Bi-sub-triple* and *Tri-sub-triple* triples capture more context and meaning. Therefore, having them at a lower priority can make the knowledge injection process prone to KN in the presence of general-purpose KG. Hence, to harness the maximum contextual knowledge from the KGs in the knowledge injection process, we have designed four distinct heuristic techniques within the *Forward Injection* method, namely *Full KG Injection Uni-sub-triple Injection*, *Bi-sub-triple Injection*, and *Tri-sub-triple Injection* to mitigate the impact of KN. These four techniques are explained below:
   a) **Full KG Injection**- When the entire KG (containing *Uni-sub-triple, Bi-sub-triple*, and *Tri-sub-triple* triples) is used as the input source for the knowledge injection, we call it the *Full KG Injection* method. In this method, the triples are injected in the order as follows: *Uni-sub-triple*, *Bi-sub-triple*, and then *Tri-sub-triple* triples, i.e., based on increasing ***"L"*** as shown in Figure 2. Therefore, this technique of injecting triples prioritizes the triples so that first we get *Uni-sub-triple*, then *Bi-sub-triple*, and finally *Tri-sub-triple* triples for injection. Prioritizing the triples here refers to the selection of specific triples from the used KGs.
   b) **Uni-sub-triple Injection**- It is the process of injecting only the *Uni-sub-triple* triples by filtering the *Uni-sub-triple* triples from the KGs used for our experiments.
   c) **Bi-sub-triple Injection**- It is the process of injecting only the *Bi-sub-triple* triples by filtering the *Bi-sub-triple* triples from the KGs used for our experiments.
   d) **Tri-sub-triple Injection**- It is the process of injecting only the *Tri-sub-triple* triples by filtering the *Uni-sub-triple* triples from the KGs used for our experiments.

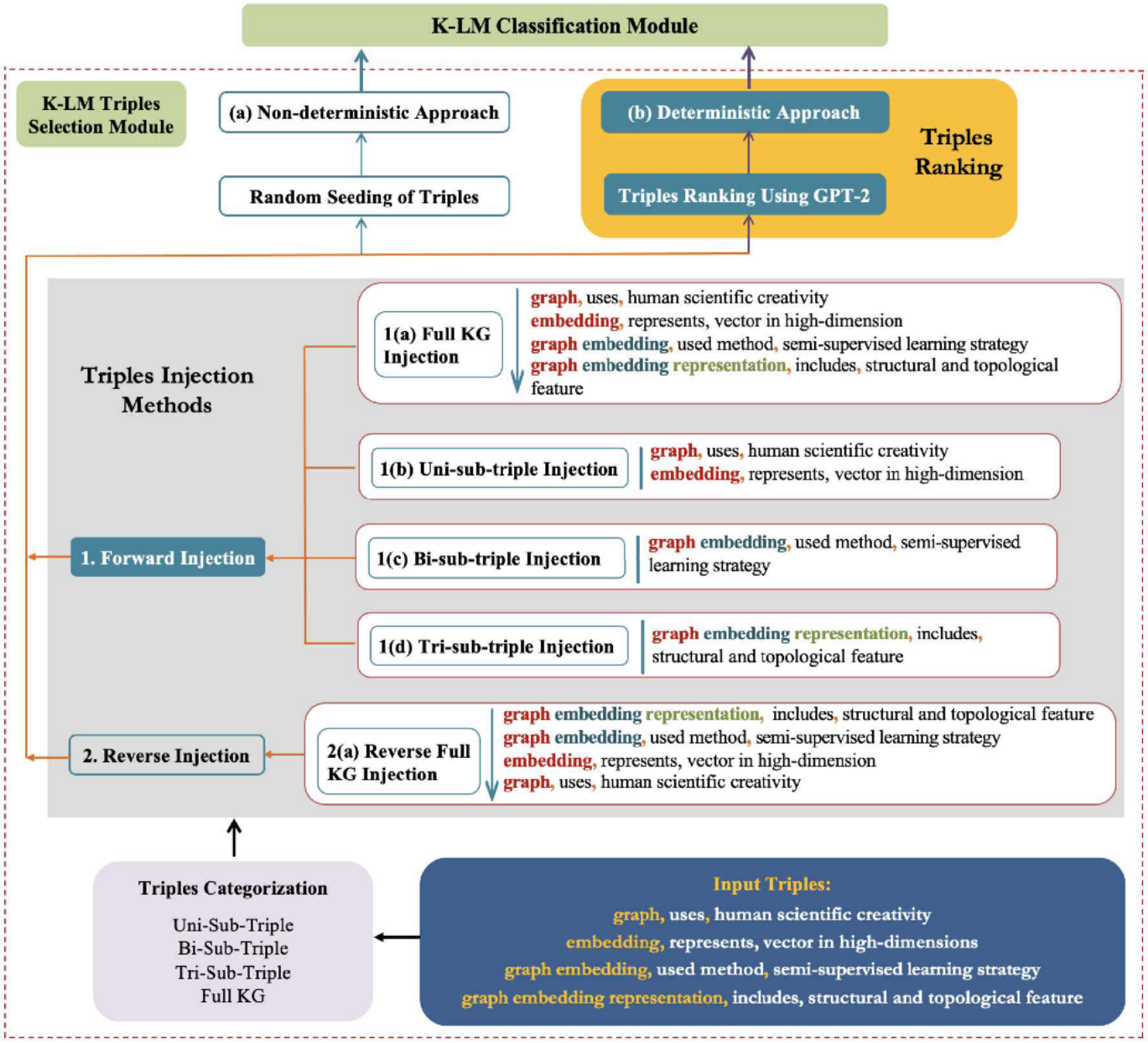

FIGURE 2. Fine-grain demonstration of K-LM Triples Selection Module. The module performs the categorization of triples and further uses them in (a) Forward and (b) Reverse Injection methods.

2) **Reverse Injection**- The opposite process of the *Forward Injection*, i.e., prioritizing the triples in the order *Tri-sub-triple*, *Bi-sub-triple*, and *Uni-sub-triple* is named *Reverse Injection*. This method takes *Reverse Full KG* as input that has exactly the opposite priority order as compared to the *Full KG*. The downside arrows in Figure 2 corresponding to *Full KG Injection* and *Reverse Full KG Injection* show the decreasing priority of the triples. The advantage of this method is that it firstly selects *Tri-sub-triple* triples, which are semantically rich in our case, for injection and thus substantially reduces the number of triples required for enhancing K-LM's performance.

Considering the examples (input sentence and the set of triples) used in the *Forward Injection* method; Figure 3 illustrates the sentence tree structure thus generated after employing the *Forward* and *Reverse Injection* methods using the *Full KG* as input triples. The vital point to note here is that *Forward* and *Reverse Injection* methods are the mandatory steps through which the triples have to pass and they are independent of the next sub-module *Triples ranking*. Depending upon the type of input triples required for knowledge injection, i.e., disordered or ordered (ranked) and are explained below.

1) **Non-deterministic Approach**- The triples received from the sub-module *Triples Injection Methods*

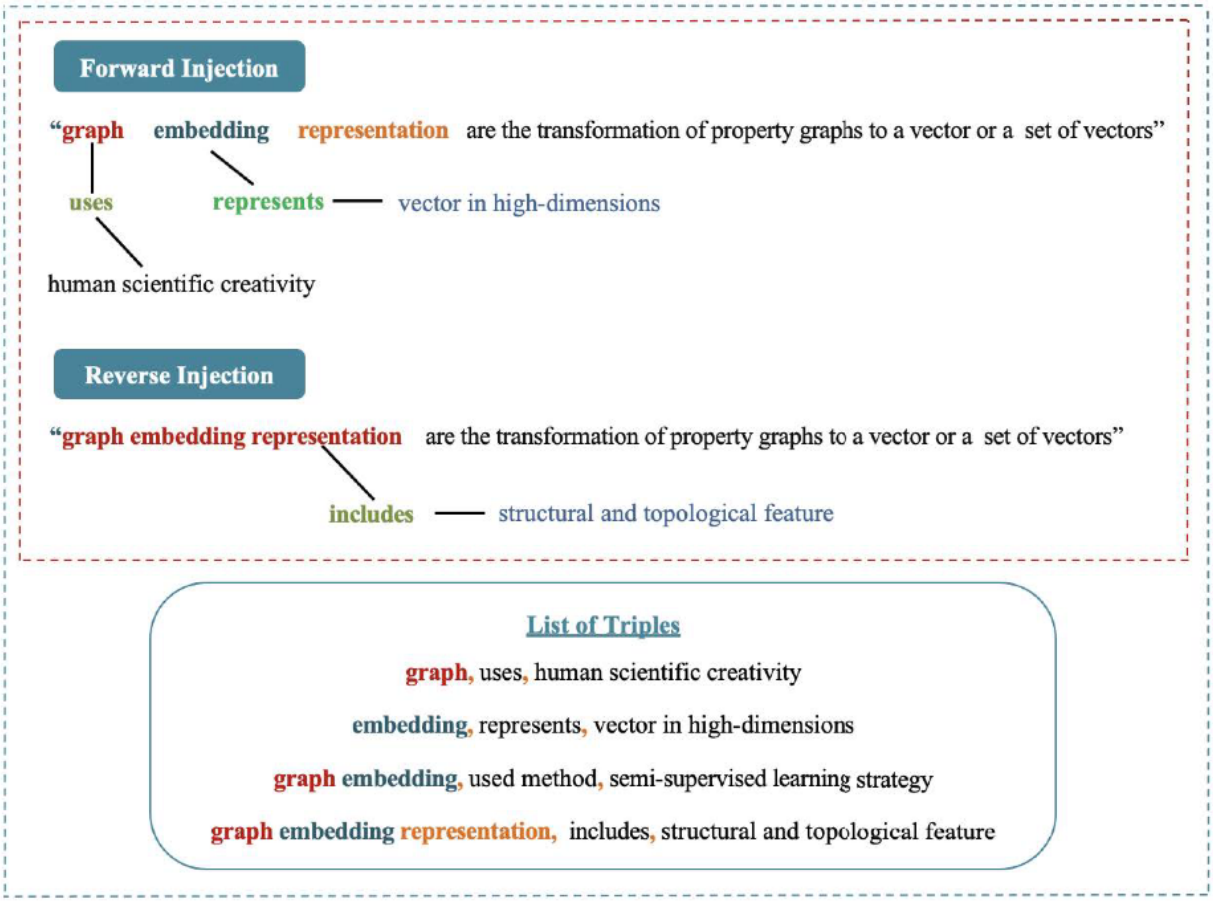

FIGURE 3. Sentence tree formation after using Forward and Reverse Injection methods for the listed triples. The priority order of triples in Forward Injection is *Uni-sub-triple* > *Bi-sub-triple* > *Tri-sub-triple* and vice-versa for the Reverse Injection.

are selected through the techniques as shown in Figure 2. At this stage, their distribution in the *look-up* table remains random and they are independent of the "context" of the input sentences. Therefore, due to the varying order of triples' arrangement, we call this random seeding of triples a `Non-deterministic` approach. In this approach, the triples are injected directly into the `K-LM Classification Module`.

2) **Deterministic Approach**- On the other hand, another possibility is to send the triples to the sub-module *Triples Ranking*. If this option is chosen, *Triples Ranking* ranks the input triples based on the probability score using GPT-2. Further, the ranked triples are organized in the *look-up* table using the ranking, and this ordered arrangement of triples is constant throughout the fine-tuning and inference stage. *Triples ranking* finds extensive uses when the experimental dataset is small and KG is sparsely related to the given dataset. We call this approach `Deterministic` and it is "context-dependent".

In other words, the above two methods can also be looked at as context-aware and context-unaware approaches to seed triples in the knowledge injection process. The rationale behind developing these methods is finding a trade-off between "relevance of KG" and "computational complexity" and maximizing the generalization of triple's injection. Hence, while the `Deterministic` approach gives an edge over the number of triples injections required to enhance the classification model's performance, randomly seeding the triples is more aligned to make the knowledge injection process independent of the dataset.

V. EXPERIMENTS AND RESULTS

The server specifications we have used to develop our methods and perform the experiments are summarized in Table 2. The `K-LM Classification Module` of K-LM takes inspiration from the LM model proposed in [32] and the source code is available at Git-hub.[16]

A. EXPERIMENTS

In this paper, we have conducted our experiments within the *Scholarly Domain* to tackle a binary classification task by using the KGs AI-KG and its variant AI-KG-Small. The experiments performed can be summed up as a combination of approaches as below mentioned:

- **Ranking the triples**- We have used `Deterministic` and `Non-deterministic` approaches to perform the experiments with $Max\_Entity \in \{1, 2, 3, 4, 5\}$.

[16] https://github.com/vsrana-ai/K-LM

TABLE 2. Server specifications.

Item	Specification
CPU	Intel Core i3-7100 (-HT-MCP-) CPU @ 3.90 GHz
GPU	NVIDIA GP102 [TITAN X], 12 GB memory
Graphic driver	NVIDIA graphic driver version 440.33.01
CUDA	Version 10.2
OS	Ubuntu (17.10)
Python	Version 3.6.6

TABLE 3. Performance of CML and DL approaches with Scholarly dataset.

Approach	Accuracy	F-1 Score
Naive Bayes (22,000 Features)	59.60	58.93
Random Forest (26,661 Features)	74.21	62.45
Bi-LSTM (Pre-Trained GloVe)	73.78	73.23
Bi-LSTM (Pre-Trained fastText)	74.88	73.48
BERT Base	78.12	75.05
K-LM using GPT-2 (Deterministic)	**81.98**	**79.55**
K-LM random seeding (Non-deterministic)	**81.78**	**79.50**

- **Forward and Reverse Injection**- We have performed the experiments with five methods of knowledge injection namely, *Full KG Injection*, *Uni-sub-triple*, *Bi-sub-triple*, *Tri-sub-triple*, and *Reverse Full KG Injection*. Here *Full KG Injection* is the default mode of seeding the triples, while the remaining four techniques proposed by us allow us to custom feed the triples.
- **Knowledge resources**- As the source of domain knowledge, we have used AI-KG and AI-KG-Small to inject the additional knowledge to perform all the experiments.
- **Bert Base**- When K-LM is not provided with any KG, then it is equivalent to a general-purpose pre-trained BERT model. Hence, in this paper, the experiment "`K-LM Without KG`" signifies the baseline approach of the BERT model.
- **Baselines**- For the baseline comparison, we have employed Naive Bayes and Random Forest classifiers as Classical machine learning (CML) and a two BiLSTM layers based network with pre-trained GLoVe[17] and fastText[18] word embeddings as DL approaches.

The performance of K-LM is measured in terms of Accuracy and F-1 score. The formulas to calculate accuracy, precision, recall, and F-1 score are given by:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{1}$$

$$Precision = \frac{TP}{TP + FP} \tag{2}$$

$$Recall = \frac{TP}{TP + FN} \tag{3}$$

$$F1 = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{4}$$

where TP, FP, and FN represent true positive, false positive, and false negative of each label, respectively.

B. EXPERIMENTAL RESULTS

In this sub-section, we present the results obtained from our experiments. To make all the employed classification approaches comparable, we have used the same distribution of test, validation, and train sets of Scholarly dataset, which are 60%, 20%, and 20% respectively. In the baseline approaches, we have tackled the problem of binary classification using CML and DL approaches which do not use any additional domain knowledge, i.e., is independent of KGs. The results of the baselines of CML and DL approaches along with K-LM are summarized in Table 3.

Tables 4 and 5, summarize the results of five `Non-deterministic` modes of triples injection with *Max_Entity* acquiring values from 1 to 5. Tables 6 and 7, summarize the results of five `Deterministic` modes of triples injection with *Max_Entity* acquiring values from 1 to 5. Finally, Tables 8 and 9 summarize the comparative distribution of $\boldsymbol{N}$, $\boldsymbol{U}$, and the actual number of triples injected for each approach used in the experiments.

C. RESULTS ANALYSIS

In this subsection, we provide insights from our results and in-depth analyses of the outcomes.

- The methods proposed to custom feed the triples into K-LM have shown superior performance and have outperformed the baselines, BERT base, and *Full KG* Injection significantly. The best performance from baselines is observed by BERT Base with an accuracy of **78.12**, while K-LM has outperformed BERT Base with all the five knowledge injection methods; the highest accuracy being **81.98** when used with *Tri-sub-triple Injection*.
- Except for *Tri-sub-triple Injection* the remaining four techniques observe a consistent decline in the model's accuracy when *Max_Entity* increases from 1 to 5. There are two reasons behind this. First, *Tri-sub-triple* triples are very less in number as compared to *Uni-sub-triple* and *Bi-sub-triple* triples, which highlight the unique association of *Tri-sub-triple* triples with the context of the input sentence. Secondly, *Tri-sub-triple* triples are semantically rich and they capture more context as compared to *Uni-sub-triple* and *Bi-sub-triple* triples. So, when more *Tri-sub-triple* triples are injected, they relate to a large part of the input sentence and enhance the semantics of the input sentence with relevant knowledge that helps the classifier to predict classes. On the other hand, injecting too many *Uni-sub-triple* and *Bi-sub-triple* triples increases the chances of introducing irrelevant knowledge along with domain knowledge in the given context and hence makes the classifier prone to KN. The KN is inversely proportional to the model's accuracy; hence, K-LM's accuracy decreases with the increase in KN. Therefore, it can be concluded that $\{Max\_Entity = 1\}$ is the optimal value for knowledge

[17] https://nlp.stanford.edu/projects/glove/
[18] https://fasttext.cc/docs/en/english-vectors.html/

TABLE 4. Performance of K-LM on the Scholarly dataset with AI-KG-Small (Non-deterministic triples seeding) using the proposed five modes of triples injection.

Experiment	Full KG Injection		Uni-sub-triple Injection		Bi-sub-triple Injection		Tri-sub-triple Injection		Reverse Full KG Injection	
	Acc.	F-1	Acc.	F-1	Acc.	F-1	Acc.	F-1	Acc.	F-1
Max_Entity= 1	74.49	71.55	**79.53**	**76.60**	**81.78**	**79.50**	**81.82**	**79.45**	**81.07**	**79.11**
Max_Entity= 2	72.74	66.90	79.23	75.60	81.13	78.85	81.77	79.05	81.05	78.80
Max_Entity= 3	71.37	66.75	77.75	73.60	81.12	78.65	81.52	78.80	80.72	77.75
Max_Entity= 4	71.22	65.60	77.44	74.00	81.02	77.60	81.45	79.45	80.85	78.40
Max_Entity= 5	70.64	63.80	76.42	72.55	80.52	78.25	81.53	79.35	80.62	72.55

TABLE 5. Performance of K-LM on the Scholarly dataset with AI-KG (Non-deterministic triples seeding) using the proposed five modes of triples injection.

Experiment	Full KG Injection		Uni-sub-triple Injection		Bi-sub-triple Injection		Tri-sub-triple Injection		Reverse Full KG Injection	
	Acc.	F-1	Acc.	F-1	Acc.	F-1	Acc.	F-1	Acc.	F-1
Max_Entity= 1	70.05	60.80	**79.68**	**76.55**	80.25	**76.90**	80.87	77.75	80.48	77.20
Max_Entity= 2	69.46	61.40	78.35	74.95	**80.28**	76.80	**81.32**	**78.50**	**80.60**	**77.30**
Max_Entity= 3	68.68	57.40	77.44	73.35	78.80	74.80	80.80	77.35	79.48	75.50
Max_Entity= 4	67.34	56.20.	76.09	72.80	78.40	74.20	80.60	77.20	79.48	75.50
Max_Entity= 5	66.28	55.45	76.10	70.95	77.90	73.60	80.40	77.05	80.12	77.20

TABLE 6. Performance of K-LM on the Scholarly dataset with AI-KG-Small (Deterministic triples seeding) using the proposed five modes of triples injection.

Experiment	Full KG Injection		Uni-sub-triple Injection		Bi-sub-triple Injection		Tri-sub-triple Injection		Reverse Full KG Injection	
	Acc.	F-1	Acc.	F-1	Acc.	F-1	Acc.	F-1	Acc.	F-1
Max_Entity= 1	75.55	69.85	**80.25**	**76.50**	**81.25**	**78.15**	**81.98**	**79.05**	80.72	**77.90**
Max_Entity= 2	72.79	67.50	79.33	75.00	80.47	77.35	81.82	78.90	**80.80**	77.40
Max_Entity= 3	72.87	67.15	78.70	75.35	80.68	77.30	80.87	77.80	80.05	77.20
Max_Entity= 4	71.92	66.80	77.19	73.50	80.60	77.20	81.68	78.55	80.58	77.40
Max_Entity= 5	71.12	64.80	76.57	72.65	80.45	76.80	81.52	78.45	79.93	76.95

TABLE 7. Performance of K-LM on the Scholarly dataset with AI-KG (Deterministic triples seeding) using the proposed five modes of triples injection.

Experiment	Full KG Injection		Uni-sub-triple Injection		Bi-sub-triple Injection		Tri-sub-triple Injection		Reverse Full KG Injection	
	Acc.	F-1	Acc.	F-1	Acc.	F-1	Acc.	F-1	Acc.	F-1
Max_Entity= 1	70.47	66.15	**80.67**	**78.20**	**80.95**	**78.50**	80.98	78.15	**81.35**	**79.20**
Max_Entity= 2	70.12	64.12	78.65	75.30	80.22	77.20	81.58	79.55	80.43	77.75
Max_Entity= 3	69.89	59.10	77.44	73.85	79.62	76.85	81.30	79.55	80.7	78.30
Max_Entity= 4	68.22	57.31	76.72	73.30	79.13	76.00	81.42	79.15	80.52	77.65
Max_Entity= 5	67.47	54.22	76.37	71.55	78.93	75.55	**81.72**	**79.25**	80.07	77.25

TABLE 8. Triples distribution of AI-KG-Small for each experiment with Scholarly dataset.

KG	Full Kg	Uni-sub-triple (Top-1000)	Bi-sub-triple	Tri-sub-triple	Reverse Full KG
AI-KG-Small (Total)	12904	2027	1719	325	2148
AI-KG-Small (Unique)	2397	113	401	82	498
AI-KG-Small (Injected)	360	113	324	66	408

TABLE 9. Triples distribution of AI-KG for each experiment with Scholarly dataset.

KG	Full Kg	Uni-sub-triple (Top-1000)	Bi-sub-triple	Tri-sub-triple	Reverse Full KG
AI-KG (Total)	1877453	37387	402845	85206	273096
AI-KG (Unique)	735884	201	35876	11248	40862
AI-KG (Injected)	16695	201	23753	7263	25575

injection when *Uni-sub-triple* and *Bi-sub-triple* triples are used.

- Our experimental findings show that in the **"quantity vs. relevance"** of triples, the relevance of triple is of utmost importance, and it directly influences the K-LM's performance. It can be understood by the fact that $\{Max\_Entity = 1\}$ allows the injection of only one triple per token of the input sentence. For such an

arrangement of input sentences in the sentence tree, the size of KGs becomes irrelevant as the performance of K-LM becomes dependent on ***U***, ***TIT*** and its relevance to the semantic context. The results mentioned in the Tables 8 and 9 validate our propositions:

a) too much knowledge injection makes the K-LM prone to KN.

b) KGs with a low N/U ratio are a better fit for knowledge injection, and so is the AI-KG-Small for our experiments.

c) *Tri-sub-triple* Triples injection requires significantly less number of triples injection, and that directly relates to less training time and resource requirement to conduct the experiments.

VI. CONCLUSION AND FUTURE WORK

Knowledge injection is a very delicate process; while too much knowledge injection makes the LM prone to noise and leads to false semantic changes, insufficient knowledge hardly improves the LM's efficacy. In this work, we propose the LM, K-LM, and a well-defined pipeline to integrate world knowledge directly in the form of triples from available world knowledge. In the pipeline, first, we have designed and implemented methods such as *Forward* and *Reverse Injection* to select and filter the triples for the knowledge integration process. In the later stage, we have employed `Non-deterministic` and `Deterministic approaches` to tackle the "relevance vs. quantity" juxtaposition. Our results show the efficacy of the proposed knowledge injection methods, as each of them has significantly outperformed the CML and DL baselines. The results demonstrate that K-LM is a potential choice to solve knowledge-driven tasks by using a few triples and helps in the presence of small training data. Our work can be extended to other domains in real-life scenarios to achieve optimal classification models' performances by only using a few triples as a source of external domain knowledge.

Since this is the first work towards quantifying the knowledge injection and implementing a pipeline to select and filter the triples, there is further scope for improvement in developing a more intelligent and optimized LM. Some limitations of the current work can be narrowed down to the manual selection of the *Max_Entity* parameter (used to control the number of triples associated per token of input sentences to mitigate the KN), the test of the K-LM for *Scholarly Domain* only, and the usage of one KG for the target domain (more KGs within the scholarly domain should be tested for a wider and more comprehensive analysis). Each of our proposed methods of injecting triples has outperformed the baseline approaches, with no method being a clear winner for all the values of *Max_Entity* $\in \{1, 2, 3, 4, 5\}$, used for our experiments. Therefore, it is an interesting future direction to automate the selection of the *Max_Entity* parameter by taking into account its dependency on the variables, such as Unique Triples of the input KG, context, and text length of input data, for an optimal knowledge injection.

To generalize the use of K-LM, the future direction of our work is also set to field-test K-LM beyond the *Scholarly Domain* in NLP downstream tasks [45]. In this regard, we target mental health and its sub-domains, which are considered complex research domains in healthcare, where the integration of external domain knowledge can lead to an increased reliability of the classification models. We also intend to use other state-of-the-art LR models such as Elmo [46], XLNet [47], etc. Further, we aim to develop novel techniques to generate KGs relevant to the available datasets of given domains to tackle more efficiently the problem of imbalanced datasets.

A. ABBREVIATIONS

Classical Machine Learning-(CML), Deep Learning-(DL), Natural Language Processing-(NLP), Bidirectional Encoder Representations from Transformers-(BERT), Knowledge Graph-(KG), Language Model-(LM), Generative Pretrained Transformer-(GPT), Language Representation-(LR), Knowledge Noise -(KN).

ACKNOWLEDGMENT

The authors gratefully acknowledge the support of NVIDIA Corporation with the donation of the Titan X GPU used for this research. Philips was not involved in the decisions related to the processing of the raw data, nor in the processing activities of the raw data.

REFERENCES

[1] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Trans. Knowl. Data Eng.*, vol. 22, no. 10, pp. 1345–1359, Jan. 2021.

[2] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," 2018, *arXiv:1810.04805*.

[3] F. Li, Y. Jin, W. Liu, B. P. S. Rawat, P. Cai, and H. Yu, "Fine-tuning bidirectional encoder representations from transformers (BERT)—Based models on large-scale electronic health record notes: An empirical study," *JMIR Med. Informat.*, vol. 7, no. 3, Sep. 2019, Art. no. e14830.

[4] L. Ehrlinger and W. Wöß, "Towards a definition of knowledge graphs," *SEMANTiCS (Posters, Demos, SuCCESS)*, vol. 48, nos. 1–4, p. 2, 2016.

[5] G. A. Miller, "WordNet: A lexical database for english," *Commun. ACM*, vol. 38, no. 11, pp. 39–41, 1995.

[6] C. Matuszek, M. Witbrock, J. Cabral, and J. DeOliveira, "An introduction to the syntax and content of Cyc," *UMBC Comput. Sci. Elect. Eng. Dept. Collection*, pp. 1–12, 2006.

[7] S. Auer, C. Bizer, G. Kobilarov, J. Lehmann, R. Cyganiak, and Z. Ives, "Dbpedia: A nucleus for a web of open data," in *The Semantic Web*. Cham, Switzerland: Springer, 2007, pp. 722–735.

[8] F. M. Suchanek, G. Kasneci, and G. Weikum, "YAGO: A core of semantic knowledge," in *Proc. 16th Int. Conf. World Wide Web (WWW)*, 2007, pp. 697–706.

[9] K. Bollacker, C. Evans, P. Paritosh, T. Sturge, and J. Taylor, "Freebase: A collaboratively created graph database for structuring human knowledge," in *Proc. ACM SIGMOD Int. Conf. Manage. Data (SIGMOD)*, 2008, pp. 1247–1250.

[10] A. Carlson, J. Betteridge, B. Kisiel, B. Settles, E. R. Hruschka, and T. M. Mitchell, "Toward an architecture for never-ending language learning," in *Proc. 24th AAAI Conf. Artif. Intell.*, 2010, pp. 1–8.

[11] D. Vrandečić and M. Krötzsch, "Wikidata: A free collaborative knowledgebase," *Commun. ACM*, vol. 57, no. 10, pp. 78–85, 2014.

[12] Q. Wang, Z. Mao, and B. Wang, "Knowledge graph embedding: A survey of approaches and applications," *IEEE Trans. Knowl. Data Eng.*, vol. 29, no. 12, pp. 2724–2743, Dec. 2017.

[13] Z. Wang, J. Zhang, J. Feng, and Z. Chen, "Knowledge graph embedding by translating on hyperplanes," in *Proc. AAAI Conf. Artif. Intell.*, vol. 28, no. 1, 2014, pp. 1–8.

[14] J. Weston, A. Bordes, O. Yakhnenko, and N. Usunier, "Connecting language and knowledge bases with embedding models for relation extraction," 2013, *arXiv:1307.7973*.

[15] S. Riedel, L. Yao, A. McCallum, and B. M. Marlin, "Relation extraction with matrix factorization and universal schemas," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics, Hum. Lang. Technol.*, 2013, pp. 74–84.

[16] M. Nickel, V. Tresp, and H.-P. Kriegel, "A three-way model for collective learning on multi-relational data," in *Proc. ICML*, 2011, pp. 1–24.

[17] M. Nickel, V. Tresp, and H.-P. Kriegel, "Factorizing YAGO: Scalable machine learning for linked data," in *Proc. 21st Int. Conf. World Wide Web*, Apr. 2012, pp. 271–280.

[18] A. Bordes, X. Glorot, J. Weston, and Y. Bengio, "A semantic matching energy function for learning with multi-relational data," *Mach. Learn.*, vol. 94, no. 2, pp. 233–259, 2014.

[19] A. Bordes, N. Usunier, A. Garcia-Duran, J. Weston, and O. Yakhnenko, "Translating embeddings for modeling multi-relational data," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 26, 2013, pp. 1–9.

[20] Y. Lin, Z. Liu, M. Sun, Y. Liu, and X. Zhu, "Learning entity and relation embeddings for knowledge graph completion," in *Proc. 29th AAAI Conf. Artif. Intell.*, 2015, pp. 2181–2187.

[21] M. Schlichtkrull, T. N. Kipf, P. Bloem, R. Van Den Berg, I. Titov, and M. Welling, "Modeling relational data with graph convolutional networks," in *Proc. Eur. Semantic Web Conf.* Cham, Switzerland: Springer, 2018, pp. 593–607.

[22] T. Trouillon, J. Welbl, S. Riedel, E. Gaussier, and G. Bouchard, "Complex embeddings for simple link prediction," in *Proc. Int. Conf. Mach. Learn.*, 2016, pp. 2071–2080.

[23] Z. Sun, Z.-H. Deng, J.-Y. Nie, and J. Tang, "RotatE: Knowledge graph embedding by relational rotation in complex space," 2019, *arXiv:1902.10197*.

[24] M. Nickel, L. Rosasco, and T. Poggio, "Holographic embeddings of knowledge graphs," in *Proc. AAAI Conf. Artif. Intell.*, vol. 30, no. 1, 2016, pp. 1955–1961.

[25] C. Shang, Y. Tang, J. Huang, J. Bi, X. He, and B. Zhou, "End-to-end structure-aware convolutional networks for knowledge base completion," in *Proc. AAAI*, vol. 33, 2019, pp. 3060–3067.

[26] D. Q. Nguyen, T. D. Nguyen, D. Q. Nguyen, and D. Phung, "A novel embedding model for knowledge base completion based on convolutional neural network," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics, Hum. Lang. Technol., (Short Papers)*, vol. 2, 2018, pp. 327–333.

[27] R. Xie, Z. Liu, J. Jia, H. Luan, and M. Sun, "Representation learning of knowledge graphs with entity descriptions," in *Proc. AAAI Conf. Artif. Intell.*, vol. 30, no. 1, 2016, pp. 2659–2665.

[28] N. Jain, J.-C. Kalo, W.-T. Balke, and R. Krestel, "Do embeddings actually capture knowledge graph semantics?" in *Proc. Eur. Semantic Web Conf.* Cham, Switzerland: Springer, 2021, pp. 143–159.

[29] F. Akrami, M. S. Saeef, Q. Zhang, W. Hu, and C. Li, "Realistic re-evaluation of knowledge graph completion methods: An experimental study," in *Proc. ACM SIGMOD Int. Conf. Manage. Data*, Jun. 2020, pp. 1995–2010.

[30] A. Rossi and A. Matinata, "Knowledge graph embeddings: Are relation-learning models learning relations?" in *Proc. EDBT/ICDT Workshops*, 2020, pp. 1–6.

[31] D. Ruffinelli, S. Broscheit, and R. Gemulla, "You can teach an old dog new tricks! on training knowledge graph embeddings," in *Proc. Int. Conf. Learn. Represent.*, 2019, pp. 1–20.

[32] W. Liu, P. Zhou, Z. Zhao, Z. Wang, Q. Ju, H. Deng, and P. Wang, "K-BERT: Enabling language representation with knowledge graph," in *Proc. AAAI Conf. Artif. Intell.*, vol. 34, 2020, pp. 2901–2908.

[33] K. M. Annervaz, S. B. R. Chowdhury, and A. Dukkipati, "Learning beyond datasets: Knowledge graph augmented neural networks for natural language processing," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics, Hum. Lang. Technol., (Long Papers)*, vol. 1, 2018, pp. 313–322.

[34] A. Rogers, O. Kovaleva, and A. Rumshisky, "A primer in bertology: What we know about how bert works," *Trans. Assoc. Comput. Linguistics*, vol. 8, pp. 842–866, Jan. 2020.

[35] P. Budzianowski and I. Vulić, "Hello, it's GPT-2—How can I help you? Towards the use of pretrained language models for task-oriented dialogue systems," in *Proc. 3rd Workshop Neural Gener. Transl.*, 2019, p. 15.

[36] A. Radford, K. Narasimhan, T. Salimans, and I. Sutskever, "Improving language understanding by generative pre-training," Tech. Rep., 2006, pp. 1–6.

[37] S. Yu, J. Su, and D. Luo, "Improving BERT-based text classification with auxiliary sentence and domain knowledge," *IEEE Access*, vol. 7, pp. 176600–176612, 2019.

[38] V. Kumar, D. R. Recupero, D. Riboni, and R. Helaoui, "Ensembling classical machine learning and deep learning approaches for morbidity identification from clinical notes," *IEEE Access*, vol. 9, pp. 7107–7126, 2021.

[39] V. Kumar, A. Verma, N. Mittal, and S. V. Gromov, "Anatomy of pre-processing of big data for monolingual corpora paraphrase extraction: Source language sentence," in *Emerging Technologies in Data Mining and Information Security*, vol. 3. Singapore: Springer, 2019, p. 495.

[40] D. Dessi, D. R. Recupero, G. Fenu, and S. Consoli, "Exploiting cognitive computing and frame semantic features for biomedical document clustering," in *Proc. SeWeBMeDA@ ESWC*, 2017, pp. 20–34.

[41] A. K. Uysal and S. Gunal, "The impact of preprocessing on text classification," *Inf. Process. Manag.*, vol. 50, no. 1, pp. 104–112, Jan. 2014.

[42] D. Dessí, R. Helaoui, V. Kumar, D. R. Recupero, and D. Riboni, "TF-IDF vs word embeddings for morbidity identification in clinical notes: An initial study," in *Proc. 1st Workshop Smart Personal Health Interfaces Co-Located With 25th Int. Conf. Intell. User Interfaces, SmartPhil@IUI*, vol. 2596, S. Consoli, D. R. Recupero, and D. Riboni, Eds. Cagliari, Mar. 2020, pp. 1–12. [Online]. Available: http://ceur-ws.org/Vol-2596/paper1.pdf

[43] D. Dessì, F. Osborne, D. R. Recupero, D. Buscaldi, and E. Motta, "Generating knowledge graphs by employing natural language processing and machine learning techniques within the scholarly domain," *Future Gener. Comput. Syst.*, vol. 116, pp. 253–264, Mar. 2021.

[44] D. Dessí, F. Osborne, D. R. Recupero, D. Buscaldi, E. Motta, and H. Sack, "AI-KG: An automatically generated knowledge graph of artificial intelligence," in *Proc. Int. Semantic Web Conf.* Cham, Switzerland: Springer, 2020, pp. 127–143.

[45] A. Bhandari, V. Kumar, P. T. Thien Huong, and D. N. Thanh, "Sentiment analysis of COVID-19 tweets: Leveraging stacked word embedding representation for identifying distinct classes within a sentiment," in *Proc. Int. Conf. Artif. Intell. Big Data Digit. Era*. Cham, Switzerland: Springer, 2022, pp. 341–352.

[46] M. E. Peters, M. Neumann, M. Iyyer, and M. Gardner, "Deep contextualized word representations," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics, Hum. Lang. Technol.*, vol. 1. New Orleans, LA, USA: ACL, Jun. 2018, pp. 2227–2237. [Online]. Available: https://www.aclweb.org/anthology/N18-1202

[47] Z. Yang, Z. Dai, Y. Yang, J. Carbonell, R. R. Salakhutdinov, and Q. V. Le, "XLNet: Generalized autoregressive pretraining for language understanding," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 32, 2019, pp. 1–11.

VIVEK KUMAR received the M.S. degree from NUST MiSiS, Moscow, Russia. He is currently pursuing the Ph.D. degree with the University of Cagliari, Italy. Currently, he is a Marie Sklodowska-Curie Researcher with the University of Cagliari, Italy, and Philips Research, Netherlands. He has authored several publications and is serving as reviewer for journals and conferences of IEEE, ACM, Springer, Elsevier, MDPI, Taylor Francis, and IGI-Global. His research interests include machine learning, deep learning, natural language processing, and sentiment analysis applied to the healthcare domain.

DIEGO REFORGIATO RECUPERO received the Ph.D. degree in computer science from the University of Naples Federico II, Italy, in 2004. From 2005 to 2008, he was a Postdoctoral Researcher at the University of Maryland College Park, USA. He has been a Full Professor with the Department of Mathematics and Computer Science, University of Cagliari, Italy, since February 2022. He is the author of more than 100 conference and journal papers in these research fields, with more than 2000 citations. His current research interests include sentiment analysis, semantic web, natural language processing, human–robot interaction, financial technology, and smart grids. In his research areas, machine learning, deep learning, and big data are key technologies employed to effectively solve several tasks. He received several awards in his career, such as the Marie Curie International Reintegration Grant, the Marie Curie Innovative Training Network, the Best Research Award from the University of Catania, the Computer World Horizon Award, the Telecom Working Capital, and the Startup Weekend. He co-founded six companies within the ICT sector and is actively involved in European projects and research (with one of his companies, he won more than 30 FP7 and H2020 projects).

RIM HELAOUI received the Ph.D. degree from the University of Mannheim, Germany, with a focus on A.I., knowledge representation and reasoning. She is a Team Member with the Data and Web Science Group, where she focused on statistical relational approaches in the context of complex human activity recognition. She is currently a Data and AI Senior Scientist with Philips Research, leading a value stream for different AI enablers in healthcare applications.

DANIELE RIBONI received the Ph.D. degree in computer science from the University of Milano, in 2007. He was a Postdoctoral Fellow and an Assistant Professor at the University of Milano. He has been an Associate Professor of computer science with the University of Cagliari, since 2015. His research interests includes activity recognition, pervasive healthcare, knowledge management, and privacy issues in pervasive, and mobile computing. He served as the TPC Chair and the TPC Vice-Chair for different conferences and workshops in the field, including IEEE PerCom and the International Conference on Intelligent Environments. His contributions appear in major conferences and journals.