Trust-Informed Large Language Models via Word Embedding-Knowledge Graph Alignment

James E. Ecker*
NASA Langley Research Center, Hampton, Virginia, 23681

B. Danette Allen †
NASA Headquarters 300 E. Street SW Suite 5R30 Washington, DC 20546

A major weakness of a Large Language Model (LLM) is its tendency to accept information at face value, often leading to injection of erroneous information and inducing a greater probability of hallucinating non-existent information. While Retrieval Augmented Generation (RAG) uses external knowledge sources to bolster LLMs through grounded truth, this work seeks to explore methods to engender a LLM with an intrinsic capability to evaluate an input's believability without relying on external knowledge sources. We investigate unifying a LLM with a Knowledge Graph (KG) and using the KG to reinforce the LLM's internal word embedding while also maintaining belief metrics along the edge's in the KG.

I. Nomenclature

General Terms and Abbreviations

AI	=	Artificial Intelligence
NLP	=	Natural Language Processing
RAG	=	Retrieval Augmented Generation
LLM	=	Large Language Model
KG	=	Knowledge Graph
KGE	=	Knowledge Graph Embedding
GNN	=	Graph Neural Network
XAI	=	Explainable Artificial Intelligence
$FAIR$	=	Facebook Artificial Intelligence Research
HMI	=	Human-Machine Interface
$NASA$	=	National Aeronautics and Space Administration
$GDPR$	=	General Data Protection Regulation

Models and Techniques

VSM	=	Vector Space Model
$Word2Vec$	=	Word to Vector model
$CBOW$	=	Continuous Bag of Words
$Skip\text{-}Gram$	=	Word2Vec architecture that predicts context words given a target word
$GloVe$	=	Global Vectors for Word Representation
$FastText$	=	FastText Word Embedding Model
$ELMo$	=	Embeddings from Language Models
$BERT$	=	Bidirectional Encoder Representations from Transformers
GPT	=	Generative Pre-trained Transformer
$TransE$	=	Translational Embedding Model
$TransH$	=	Translational Embedding with Hyperplanes
$TransR$	=	Translational Embedding with Relation Spaces
$DistMult$	=	DistMult Bilinear Model
$ComplEx$	=	Complex Embeddings for Link Prediction

*Research Computer Scientist, Autonomous Integrated Systems Research Branch, NASA Langley Research Center, Hampton, VA
†SCL for Autonomous Systems, NASA HQ

$RotatE$ = Rotational Knowledge Graph Embedding
$RGCN$ = Relational Graph Convolutional Networks
$KGAT$ = Knowledge Graph Attention Networks
$KAGNet$ = Knowledge-Aware Graph Networks
$ERNIE$ = Enhanced Representation through Knowledge Integration
$K-BERT$ = Knowledge Integration BERT
$KnowBERT$ = Knowledge-Enhanced BERT
$COMET$ = Commonsense Transformers for Knowledge Graph Construction
$KEPLER$ = Knowledge Embedding and Pre-trained Language Representation
$LLAMA$ = Large Language Model Meta AI
DMN = Dynamic Memory Network
PSL = Probabilistic Soft Logic
$FinBERT$ = Financial-domain adaptation of BERT

Mathematical Symbols

E = Set of entities in a knowledge graph
R = Set of relations in a knowledge graph
(h, r, t) = A triple consisting of head entity h, relation r, and tail entity t
$\mathbf{h}$ = Embedding vector of the head entity
$\mathbf{r}$ = Embedding vector of the relation
$\mathbf{t}$ = Embedding vector of the tail entity
$f(\cdot)$ = Mapping function or transformation
L = Loss function
θ = Model parameters
$\mathcal{G}$ = Knowledge graph represented as a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$
$\mathcal{V}$ = Set of vertices (entities) in a graph
$\mathcal{E}$ = Set of edges (relations) in a graph
$P(h, r, t)$ = Probability score assigned to a triple (h, r, t)
$\mathcal{D}$ = Dataset or corpus used for training

Metrics and Evaluation

AUC = Area Under the Curve
MRR = Mean Reciprocal Rank
$Hits@K$ = Proportion of correct entities ranked in the top K positions
F_1 = Harmonic mean of precision and recall

Other Symbols

X = Input data or features
Y = Output data or labels
$\mathbb{R}^n$ = n-dimensional real space
$\mathbb{C}^n$ = n-dimensional complex space
$\sigma(\cdot)$ = Activation function (e.g., sigmoid, ReLU)
∇ = Gradient operator
α = Learning rate in optimization algorithms
T = Set of triples in a knowledge graph

II. Introduction

Large language models such as GPT3/4, LLAMA, and BERT have revolutionized human-machine interface (HMI) and Natural Language Processing (NLP) research. These new tools afford machines an unprecedented ability to comprehend and produce human level natural language text. As such, LLMs have been deployed in various fields of application such as chatbots, content creation, machine translation, and multi-modal text-to-image/video generation. Despite these impressive capabilities, researchers are presented with significant challenges in trustworthiness of both input prompting and generative outputs of the models under consideration. The latter has become described by practitioners as "hallucinations," generative output that is not grounded in truth.

Critically, hallucinations in LLMs pose a barrier of uncertainty with respect to their deployment in safety and fact critical applications such as the aerospace, healthcare, legal, financial fields. For instance, deployment of a hallucination-prone LLM in aerospace applications could lead to decision making being made based on unreliable generative data that does not reflect the state and environment of the physical system, compromising mission safety. This issue arises from how the LLM is trained. LLM training relies on statistical patterns in large, general subject natural language text corpora. The statistical nature of the training is what enables the model to generate text that believably reflects the patterns found in the training data, and there exists no factual grounding in this process beyond that which is reflected in the text sequence distribution of the training corpora itself.

A. Limitations of Current Approaches

To address the problem of hallucinations, researchers have investigated techniques such as Retrieval Augmented Generation (RAG), wherein large language models make use of supplementary knowledge from external sources to vary and enrich their outputs. Although varying and enriching the output does enhance the perceived accuracy of the model, it also creates some new problems for us to solve. As good as RAG is, it isn't the panacea for all our hallucination-related concerns. When we do depend on it, we introduce several new points of potential failure, most notably (1) delays in response time and (2) dependence on external networked systems that could fail at any time. Additionally, RAG methods do not change the LLM's inherent way of determining whether or not something is believable. Instead, they function as an external fix, treating the disease's symptoms but not the root cause. What we need are techniques that give LLMs the power to assess information's plausibility on their own, with no need for over-reliance on outside systems.

B. Objective of this Work

This paper seeks to investigate contemporary approaches for aligning word embeddings with knowledge graph embeddings and how they can be used to enhance the trustworthiness of the generative outputs of the LLM. Our concentration, in particular, is on the following:

1) **Surveying Current Techniques**: Providing a comprehensive overview of methodologies for integrating KGs with LLMs, including mapping-based methods, joint embedding techniques, and the use of graph neural networks.
2) **Discussing Applications**: Examining how these alignment techniques can be applied to reduce hallucinations, improve information verification, and enhance the believability evaluation capabilities of LLMs.
3) **Identifying Challenges and Future Directions**: We highlight the technical, methodological and ethical challenges in this domain and suggest potential avenues for future research.

As such, this work seeks to be a valuable resource for those interested in developing AI systems that are more reliable and trustworthy by serving the reader in two capacities: first, as a tutorial for those who might not be familiar with the area; and second, as a survey that highlights the current state of research and practice.

C. Organization

This paper is organized as follows:

1) **Section 3: Fundamentals of Word Embeddings and Knowledge Graphs** We introduce the concepts of word embeddings, knowledge graphs, and knowledge graph embeddings to develop a foundational understanding of their integration.
2) **Section 4: Alignment of Word Embeddings and Knowledge Graph Embeddings** Here we describe our key prospective methodology for LLM/KG integration — embedding alignment — and explore the challenges associated with using it.
3) **Section 5: Survey of Current Methods** This section provides a review of existing techniques and case studies in improving LLMs using KG.
4) **Section 6: Applications Leading to Trust-Informed LLMs** We discuss how the methods surveyed contribute to the development of LLMs that are amenable to intrinsic believability evaluation.
5) **Section 7: Challenges and Open Issues** We address technical, data-related, methodological, and ethical challenges in aligning embeddings and suggest solutions where possible.
6) **Section 8: Conclusion** We summarize the key points and reflect on the implications of embedding alignment for the future of trustworthy AI.

In summary, the challenge of hallucinations in LLMs underscores the critical need for models that can intrinsically assess the believability of information. This paper aims to contribute to the development of more trustworthy AI

systems by surveying and synthesizing contemporary approaches for aligning word embeddings with knowledge graph embeddings. We believe that embedding alignment holds significant promise in enhancing the reliability of LLMs without over-reliance on external systems. We invite readers to explore the following sections, which provide both foundational knowledge and insights into cutting-edge methodologies that could shape the future of NLP and AI trustworthiness.

III. Fundamentals of Word Embeddings and Knowledge Graphs

A. Word Embeddings

Word embeddings are vector representations of words in a continuous, low-dimensional space where semantically similar words are mapped to proximate points. They are fundamental in enabling machines to process and understand human language by capturing syntactic and semantic relationships between words in latent space. The NLP field has seen the revolutionary impact of word embeddings, as they serve as a way to represent words in a form that is easily digestible for machines. Traditionally, language processing approached the problem using discrete representations that were not well understood by today's standard of machine learning. For instance, one-hot encoding treats words as isolated. It does not capture the relationships between words, nor does it appear to identify the meaning of the words themselves. In contrast, word embeddings allow for a more nuanced representation of language using the vector space model (VSM). Word embeddings' main advantage is their ability to capture semantics and syntax. For example, when the embedding space is well trained, words with closely related meanings (like "king" and "queen" or "car" and "automobile") cluster together [1]. This proximity isn't coincidental; it shows how well word embeddings reflect the relationships between words and the contexts in which they appear. Word embeddings can even encode more complex relationships like analogies. A classic example is the vector arithmetic operation: "king" - "man" + "woman" = "queen." This operation demonstrates not just the power of word embeddings to capture individual word meanings but also their power to capture the relational dynamics between words.

The consequences of word embeddings go deeper than simple representation; they are key elements in many natural language processing applications, including sentence analysis, machine translation, and information retrieval, to name a few. By serving as a bridge from the human world to the machine world, they make possible the kind of quantification required to perform linguistic tasks at scale.

1. Traditional Embeddings

Traditional word embeddings assign a single static vector to each word, irrespective of context. Typically, word embeddings are formed by training on an algorithm such as those listed below. These methods use extensive text corpora to derive word representations from their local contexts.

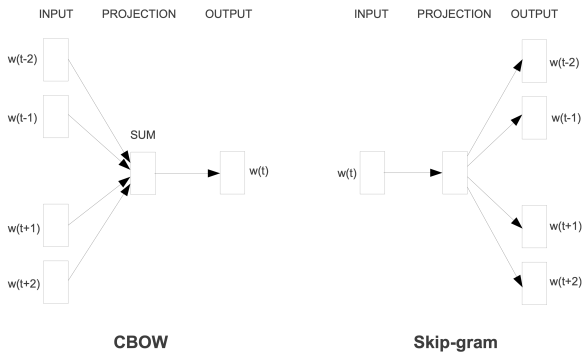

Fig. 1 Word2Vec architectures Mikolov *et al.* [2]

1) **Word2Vec**[2] is a neural network model devised for the purpose of producing word embeddings, which are compact vector representations of words. These word embeddings carry the meaning and relationships of the words in a continuous vector space and are used in many natural language processing tasks. To put it simply, Word2Vec looks at a large body of text and learns the properties of words based on the way they are used. It pays

attention to how often words are used together and in what contexts. It comes in two architectures, shown in Figure 1:

1) **Continuous Bag-of-Words (CBOW)**: Predicts the current word based on the surrounding context words. This architecture aims to use context to accurately predict which word should occur in a certain position of a sentence. The neural network acts as a virtual linguist to determine which word might best fit the blank when given the kinds of context words on either side of it that would normally be found in a well-formed English sentence. In the case of CBOW, "virtual linguist" translates to "set of probabilities over the vocabulary."
2) **Skip-Gram**: Predicts surrounding context words given the current word. In contrast, the Skip-Gram model operates by forecasting the words that bring meaning to a sentence around a given target word. For instance, if the target word is "king," the model's job is to determine what contexts word in the model's knowledge base might work with "king" to create a semantically coherent unit of meaning. Context words like "royal," "throne," or "monarch" are all different angles on the kind of world in which "king" is a meaningful word. These different angles correspond to the basic idea of capturing the relationships between words. Skip-Gram is particularly good at that, and its ability to do so serves the model well in generating high-quality embeddings.

Word2Vec captures linear relationships between words, enabling operations like $king - man + woman \approx queen$. This linear relation demonstrates that the Word2Vec model understands not just the meaning of individual words but also how they relate to each other. And when we say relate, we mean in a way that allows for what mathematicians or linguists might call "algebraic operations" in the "vector space" that Word2Vec creates. Such operations, or at least their results, show off the marriage of structure and meaning that is a hallmark of natural human language.

2) **Global Vectors for Word Representation (GloVe)** [3] combines global matrix factorization and local context window methods. It constructs a global word-word co-occurrence matrix from a corpus and factorizes it to obtain word vectors. The main advancement of GloVe, compared to earlier types of word embeddings, is that it can encode both local and global statistical information into its word vectors. Local context window methods typically focus on the immediate neighbors of a word. GloVe, on the other hand, considers the overall distribution of word co-occurrences throughout the entire corpus and uses this global information in tandem with local information to produce word vectors with a high degree of semantic fidelity.
3) **FastText** [4], developed by Facebook's AI Research (FAIR) lab, is an extension of the Word2Vec model that addresses some of the limitations associated with traditional word embedding techniques by representing words as bags of character n-grams. In their research, the authors explain how this simple innovation has profound effects on word representations. By using n-grams, which are sequences of n characters, the model can capture information about the spelling and structure of words. This is especially useful for languages with rich morphology (like Arabic or Finnish) and for dealing with rare or out-of-vocabulary words. In the classic Word2Vec model, every word is viewed as an indivisible unit. The model generates a static embedding for each word based solely on the context in which the word appears. This method works well with frequent words, yet it has problems with rare words and words excluded from the training data. FastText improves this situation by looking at smaller parts of the words when forming the word embeddings. The parts that FastText looks at are the character n-grams. For example, to form the embedding for the word "embedding," FastText looks at the following parts of the word: "em," "emb," "bed," "din," and "ing." When FastText forms the word embedding for "embedding," it partially aggregates the embeddings of these character n-grams.

   The morphological richness of a language and the number of forms a single word can take can significantly affect the performance of word embeddings. Many models learn effective representations only for the words (i.e., surface forms) they have seen during training. Morphologically complex languages, however, have many kinds of variation that a model can never fully see or capture. But if a model somehow knows the structure of a surface form, it can use that knowledge to generate the representations of all the other surface forms that the model hasn't seen. FastText is one model that can do this. It predicts representations of words by first representing the word as a bag of n-grams.

   In addition, the capability to create embeddings for out-of-vocabulary words boosts FastText's relevance in real-life situations where we often see newly minted terms, like in social media or when people are using specialized, technical language. This flexibility makes the model's performances not only better in certain respects but also adds something interesting to the whole picture of what the model is achieving in terms of understanding language.

All of these methods attempt to represent words in such a way that semantic relations are preserved between them.

2. Contextualized Embeddings

Unlike traditional word embeddings, which yield a fixed representation for a word regardless of its appearance in a sentence, contextualized embeddings vary in accordance with the semantic demands of the specific use of the word, within a sentence or even across sentences. For instance, in the sentences "I went to the bank to deposit money" and "I sat by the bank of the river," the different uses of "bank" in this pair of contexts would elicit different word vectors because the meaning of "bank" is not the same in these two instances. Generating representations that are sensitive to the context not only handles word meaning disambiguation, but also improves the model's grasp of syntactic and semantic relationships among words in a sentence. This is very beneficial for tasks like sentiment analysis, machine translation, and question answering, where the context is everything and the slightest shift in situation can (and sometimes does) produce wildly varying responses. Moreover, the performance of different NLP baselines has improved dramatically with the introduction of contextualized embeddings. These embeddings have proven to be very effective in understanding the nuanced structure of human language and its many surface forms. However, the real power of these architectures lies in their ability to compress the vast amount of information contained in human language down to the number of parameters in the model without resorting to hard-wired linguistic rules. At the same time, the models retain the plasticity necessary to learn in different contexts. Contextualized embeddings generate dynamic word representations that vary depending on the word's context within a sentence, addressing polysemy and contextual nuances. Significant examples of contextualized embeddings include:

1) **Embeddings from Language Models (ELMo)** [5] In contrast to conventional word embeddings that provide a fixed vector for each word irrespective of context, ELMo goes a step further by generating dynamic representations of words—based on the complete input sentence—that are context-sensitive. This inherently flexible approach not only permits ELMo to capture the full diversity of semantic word meanings, but also to take into account the syntactic variability of words when jumping from one context to the next.
   ELMo's architecture is based on a bidirectional LSTM. Elmo first encodes the input text in the forward direction and then in the backward direction. The two sets of LSTM cell states are concatenated and passed to a dense layer to produce the final output at each time step. The result is a layer of character-based embeddings that can effectively capture the kinds of semantics and syntax that make human languages nuanced and rich (context!). In addition, the hidden states of the LSTM, particularly the multiple layers of the network, generate ELMo embeddings. This approach of using different layers enables ELMo to grasp different levels of abstraction. It can "see" the words in a sentence in various ways, from the simplest surface-level level to the kinds of semantic relationships that you or I might think about when we read a sentence. As a result, ELMo has been shown to improve the performance of a wide array of NLP tasks, including sentiment analysis, named-entity recognition, and question-and-answering. On these kinds of tasks, the models using ELMo do much better than they do when using static representations.

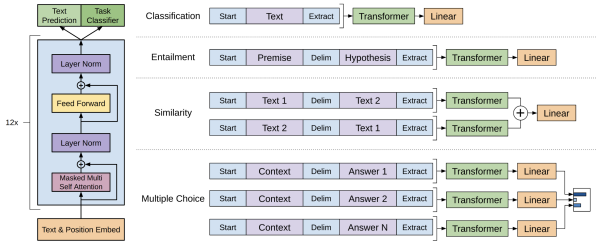

Fig. 2 Transformer architecture and input transformations in GPT Fig 1. Radford *et al.* [6]

2) The **Generative Pre-trained Transformer (GPT)** models [6–8] are unidirectional transformer-based architectures pre-trained to predict the next word in a sequence, as shown in Figure 2. The unidirectional architecture that underpins language models like GPT-3 and GPT-4 means that, during training, the model bases its predictions of the next word in a sequence on the context that comes right before it and not on any context that comes after it. Besides being a realistic first step toward producing a truly intelligent conversational partner, unidirectional training has the added advantage of making the language model easier to use with different input lengths from different users.

Training a GPT model means showing it a huge amount of text data, and from this, the model learns to recognize the complex patterns, the subtlety, and the very structure of human language. The core of the performance of these models lies in how well they scale. With tens of billions of parameters, GPTs can deal with the sort of scale that lets them capture the kind of intricate relationships within the data that makes their outputs not just grammatical but also contextually sensible. But scaling up brings important worries along with it—from ethical considerations of how the models will be used when they can generate text at human-like speed to concerns over the potential for generating texts that might mislead people or reflect the biases of the people who put the training data together in the first place.

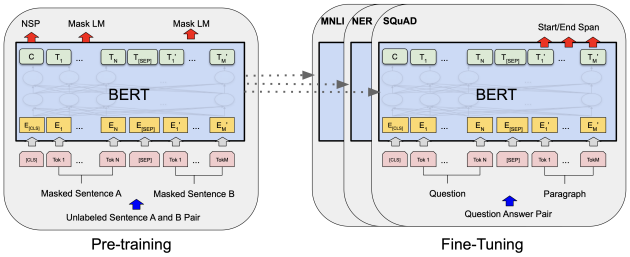

Fig. 3 BERT model pretraining and fine tuning procedures Fig 1. Devlin *et al.* [9]

3) **Bidirectional Encoder Representations from Transformers (BERT)** [9] a transformer-based model, was developed by researchers at Google. Figure 3 shows its unique training approach—masking part of the language (text) during training and predicting what is missing (which word or words could possibly fit there) and also predicting the next sentence (or part of the next sentence)—allows BERT to do something most language models cannot: deeply understand the language and its nuances.
BERT's masked language modeling requires that a portion of the input tokens be randomly masked. When the model is run, these tokens need to be predicted from the context in which they were seen. Because BERT is trained to do this very well, it does not just learn a trivial association of a word with the slots where it can or cannot be; it learns a compact representation of the meaning of a word from the context of the rest of the sentence.
Having both the preceding and subsequent contexts, BERT is effectively producing bidirectional embeddings, which are generally better than unidirectional ones, especially for tasks that require a more nuanced understanding of the word's meaning.
Moreover, the task of predicting the next sentence also significantly boosts BERT's comprehension of the relationships between sentences. When one trains a model to predict whether one sentence follows logically after another, one is making the model pay close attention to the coherence of the two sentences in context. This kind of attention is necessary for many NLP applications and is very relevant to the architecture of BERT.

In summary, word embeddings are a supremely useful tool in the NLP realm. They allow a computer to go beyond just seeing the words of an unprocessed text and to "understand" at some level the real semantics and the abstract meaning of the text. Seeing at least a couple of different dimensions through which to view the tool helps to appreciate how it works and how to use it. The two directions, neural networks and matrix factorization, each lead to powerful, state-of-the-art applications.

B. Knowledge Graphs

Knowledge graphs (KGs) are structured representations of real-world entities and their interrelations, encoded as a collection of triples in the form $(head\ entity, relation, tail\ entity)$. KGs provide an avenue for representing information that is both human-readable and comprehensible to machines. Entities, relations, and attributes serve as the fundamental building blocks of KGs; together, they provide a true understanding of the interconnectedness of real-world concepts.

1. Definition and Components

A knowledge graph consists of:

1) **Entities (Nodes)**: Objects or concepts in the domain, such as people, places, or events. These entities may be

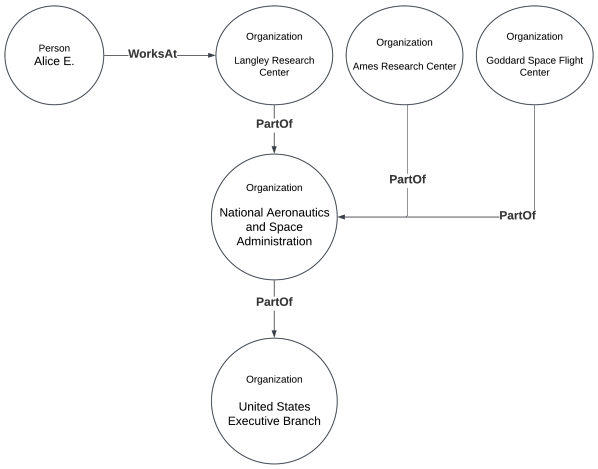

Fig. 4 Example of simple knowledge graph showing entity-relationships

anything from palpable items like *New York City* to personages such as *Albert Einstein*. More often than not, they also embody the types of events or occurrences that can be part of a graph, like *The Invention of the Airplane*. Each entity typically gets its own unique key or pointer for easy referencing.

2) **Relations (Edges)**: Directed edges representing the relationships between entities. They help explicate the nature of the various connections and interactions that entities have. For example, the triple (*Alice*, *worksAt*, *National Aeronautics and Space Administration (NASA)*) represents the fact that Alice is employed by NASA. Figure 4 shows a portion of a toy example knowledge graph with a directed edge representing the *worksAt* relation that connects the entity *Alice* with the entity *NASA*. This type of directed edge is great for illustrating point-to-point associations—for employment in this case, but also potentially for any number of associative, hierarchical, or temporal kinds of relationships
3) **Attributes (Properties)**: Attributes offer extra context and descriptiveness about entities or relations. For instance, the entity "Alice" might possess attributes such as *age*, *educational background*, or *expertise*, while the relationship *worksAt* could include attributes like *starting date* or *position title*. Attributes enhance the knowledge graph by supplying it with additional detail that can be key to comprehending the nature of the entities and their relationships. This extra context can be especially valuable in applications like data analytics, where the main focus is on deriving insights from both the relationships and the properties of the involved entities.

2. Examples of Knowledge Graphs

1) **Wikidata** [10] is a collaboratively edited knowledge base operated by the Wikimedia Foundation. It serves as a central storage for the structured data of its Wikimedia sister projects, including Wikipedia.
2) **DBpedia** [11] extracts structured content from the information created in the Wikipedia project and provides a public data set for semantic Web and linked data applications.
3) **Freebase** [12] was a large collaborative knowledge base consisting of data composed mainly of its community members. It was acquired by Google and contributed to the development of Google's Knowledge Graph [13].

C. Knowledge Graph Embeddings

Knowledge graph embeddings map entities and relations in a KG to continuous vector spaces while preserving the graph's structural and semantic information. These embeddings facilitate the application of machine learning algorithms to KGs by converting all the entities and relationships in a KG and into vector space representations. This allows us to use different linear algebra techniques that are very low in computational complexity, that is, they are fast and efficient. We can use these mathematical techniques to derive a number of different algorithms, and the performance of these algorithms can be significantly improved.

1. Purpose and Applications

The primary purposes of KG embeddings include:

1) **Link Prediction**: Inferring missing relationships between entities. Link prediction is among the most important applications of knowledge graph embeddings. In many real-world cases, knowledge graphs are incomplete, that is, they do not represent all existing relationships among entities. There are also many potential links that have yet to be documented. Machine learning models leverage the embeddings not only to analyze the patterns and similarities among the existing entities but also to use these entities as a sort of "context" for the otherwise ambiguous existing relationships. In a social network knowledge graph, for instance, two users with no known connection might nevertheless be predicted to "know" each other by virtue of their sharing many graph-like features, such as interests and connections to the same entities.
2) **Entity Classification**: Assigning categories or types to entities based on their embeddings. By directly linking entities to a continuous vector space, we can leverage the interacting characteristics of those entities to more easily identify and categorize them. This is directly beneficial in situations where we know entities belong to multiple categories or types. An example is a KG that represents the world of academic publications; in it, an embedding directly benefits the classification of papers into either one or many research domains, particularly useful when the paper as exhibited by the entities it contains could qualify for several types, as is often the case.
3) **KG Completion**: Enhancing the KG by predicting new facts to fill in incomplete areas, making it ever more complete and useful. With embeddings, we can generate plausible entities and relationships that fit with the existing structure of the graph.

Knowledge graph embeddings have several key advantages relative to traditional tricks of knowledge representation. They allow for efficient computation, as most practical operations in vector spaces are less expensive (both in terms of time and space) than operations in discrete spaces. They also pack in much more information. KG embeddings represent entities and relationships as vectors in a space where both the structure and the semantics of the knowledge graph are captured. This means that KG embeddings allow us not only to do reasoning tasks efficiently but also to learn efficiently from the knowledge graph. In addition, KG embeddings promote transfer learning. The knowledge obtained from one task is readily applicable to another task. This is particularly beneficial when dealing with several KGs or when transitioning to a new domain. The embeddings can also be fine-tuned to account for changes in structure or meaning that might arise from using another graph. Overall, they present an incredibly versatile tool in AI.

Overall, embedding knowledge graphs is a powerful way to represent entities and relationships in a continuous vector space, which makes KGs much more suitable for application with machine learning algorithms. The reason for this is pretty straightforward: the algorithms that we usually train and test have a difficulty level associated with them; if we can represent something in a space where two instances that are similar should ideally be close together and two instances that are not similar should be further apart, then we have a good chance of the algorithm being able to reason with the representations and do what we want it to do, which usually involves some form of classification or regression.

2. Common Techniques

Several models have been proposed for learning KG embeddings:

1) **Translational Embedding (TransE)** [14] models relationships as translations in the embedding space. For a triple (h, r, t), it aims to ensure that $\mathbf{h} + \mathbf{r} \approx \mathbf{t}$, where $\mathbf{h}$, $\mathbf{r}$, and $\mathbf{t}$ are the embeddings of the head entity, relation, and tail entity, respectively.

   TransE's primary goal is to guarantee that the embedding of the head entity, when acted upon by the embedding of the relation, gives an approximate result, that is, a very close result—that is not identical but very close to the tail entity. In other words, the embedding of the head entity plus the embedding of the relation approximately equals the embedding of the tail entity. The approximation means that whatever the heuristic and the remainder term count up to, they should essentially give us something that looks like a truth statement or an equation that expresses a valid relationship. TransE's translation-based method effectively captures the semantics of the relationships in the knowledge graph.

   TransE's uncomplicated design is one of its strong points; it allows the model to compute quickly and scale easily. Still, it has its weak areas. TransE encounters trouble when it works with complex relationships or when the same entities are associated with several different relations. Despite these setbacks, TransE serves straightforwardly and well as a foundation model for knowledge representation. It has inspired several subsequent models, most of which build on the idea of using translations in the embedding space as a way to represent knowledge.
2) **TransH** [15] is an extension of the TransE model designed to enhance the representation of entities in knowledge

graphs by addressing the limitations of TransE in handling complex relationships.
TransE represents entities and relations as vectors in a space with continuous values. However, it assumes that an entity has just one embedding, which is used everywhere the entity is mentioned. This can cause a lot of inefficiencies for TransE, especially in situations where an entity is used in more than one relation and a different embedding is needed for the entity in the relation it is actually in.
Hyperplanes corresponding to relations serve as projection spaces for entities within those relations in TransH. The idea is that, for each relation, there is a way to represent the entities that are connected by that relation such that the entities' embeddings make sense given the relation. Specifically, when an entity is in relation to something else, its embedding is projected onto a hyperplane defined by that relation.
Projecting entities onto hyperplanes that are specific to relationships is especially useful for dealing with different kinds of associations, such as one-to-many, many-to-one, and many-to-many relationships. Let's take the entity "Paris" as an example. "Paris" might be associated with various relations, like "is the capital of," which would yield a unique representation, and "is a city in," which could be represented differently, depending on the context of any number of countries. Because of such distinct embeddings, we might say that TransH is better at modeling the knowledge graph that's associated with "Paris."
In addition, hyperplanes enable the model to concentrate on the learning of the relation, separating head and tail entities in such a way that it is minimally dissimilar to all other possible triples. If a model learns to use such hyperplanes effectively, then it must also learn to represent the graph in such a way that all or most of the edges are represented similarly. Therefore, hyperplanes can be thought of as not only improving the model's ability to learn relations but also enriching its capacity to learn the overall graph structure.

3) **TransR** [16] models entities and relations in distinct embedding spaces. It introduces a projection matrix for each relation, enabling more flexible representations of complex relations. TransE embeds entities and relations in the same vector space, and this can lead to difficulties when it comes to representing relationships—especially complex ones. TransR, on the other hand, keeps its embeddings for entities and for relations in separate spaces. This leads to a more refined set of translations across the range of relationships that exist in our world—especially those relationships that defy simple geometric interpretations.
The main improvement brought in by TransR lies in the use of a projection matrix for each relation. With the help of this matrix, one can project entity embeddings from the entity space into the relation space. When doing so, it is very important to remember that TransR is not just trying to find a single matrix that will work for all relations. Instead, it finds a different, relation-specific projection matrix for each of the N relations in the knowledge base. Complex relationships involving multiple entities or elaborate interactions are better served by this approach. The reason is straightforward: if entities can be represented in a space tailored to each specific relation, then the overall representation will necessarily be better for the diversity of relationships that appear in knowledge graphs. This flexibility should also help with various model performance metrics—link prediction, for example—that are served by having richer and more context-saturated embeddings.
4) **DistMult** [17] simplifies its bilinear scoring function by using diagonal matrices, effectively modeling symmetric relations but struggling with antisymmetric ones. The use of diagonal matrices limits the model's ability to capture antisymmetric relationships. Antisymmetric relations are those in which the association between two entities is not mutual. For example, if entity A is related to entity B, then entity B is not related to entity A in the same way. The diagonal matrix structure of DistMult makes it inherently unsuitable for modeling such relations. This is because it can't do anything with antisymmetric relations that it wouldn't also be doing with symmetric relations, since the structure has a kind of built-in symmetry to it.
5) **Complex Embeddings (ComplEx)** [18] extend DistMult into complex number space, allowing the model to capture both symmetric and antisymmetric relations by leveraging the properties of complex conjugates. Using complex numbers enables the representation of a wider variety of relational patterns, including many that are common in real-world data, and allows for the representation of both symmetric and antisymmetric relationships. Embedding models such as DistMult represent entity relationships as real-valued vectors. This works for a lot of data. But when you get to certain relational types—like antisymmetry, where the relationship between two entities is fundamentally different depending on which direction you go—bottom line, they don't work so well. And why? Because you are using a structure that essentially represents everything in some sort of "space." In this "space," all vectors are oriented in the same way when it comes to basic relational properties, like symmetry. The limitation is dealt with by ComplEx, which embeds entities in complex vector spaces and represents relations as complex matrices. What motivates this choice? For many reasons, complex numbers are just more powerful. They allow you to take advantage of the properties of complex conjugates, necessary for telling apart symmetric

and antisymmetric relations. Moreover, complex entities have a "natural" inner product, in the sense that it accommodates both symmetric relations (like father and mother) and antisymmetric relations (like husband and wife), as well as many other relations, without distinction or special cases built into the model.
The ComplEx model can be expressed mathematically as follows:
The equation that expresses the force as a function of the magnetic field h, position r, and time t is given by:

$$f(h, r, t) = \text{Re}(\mathbf{h} \cdot \mathbf{r} \cdot \mathbf{t}^*)$$

This shows how these three factors interact to create a force.
In this representation, **h**, **r**, and **t** stand for the head entity, the relation, and the tail entity, respectively; **h** and **t** are their corresponding complex embeddings; and **r** is the complex representation of the relation. The symbol $\cdot$ indicates the inner product, and $\mathbf{t}^*$ signifies the complex conjugate of the tail entity's embedding. The model capitalizes on the real part of this product to effectively navigate the intricacies of the relationships and to capture the relevant interactions.
Modeling both symmetric and antisymmetric relations allows the ComplEx model to express nuances in knowledge graphs that are critical for tasks like knowledge graph completion and link prediction. Why is this? Complex numbers allow for richer mathematical manipulations, and this allows the model to learn richer (or "more expressive") embeddings.

6) **RotatE** [19] represents relations as rotations in the complex vector space. It models various relation patterns, including symmetry, antisymmetry, inversion, and composition, by enforcing that

$$\mathbf{t} = \mathbf{h} \circ \mathbf{r}$$

where $\circ$ denotes the Hadamard (element-wise) product. This approach allows for a nuanced representation of various relational patterns, which can be critical for tasks such as link prediction and knowledge graph completion.
The model can guarantee that if $(\mathbf{h}, \mathbf{r}, \mathbf{t})$ is a valid triplet, then $(\mathbf{t}, \mathbf{r}, \mathbf{h})$ is also a valid triplet, provided, of course, that the relationship is symmetric. This is done by ensuring that the rotation required to go from **h** to **t** is equal to the rotation required to go from **t** to **h**. In this case, the rotation required to go from **h** to **t** is denoted $\mathbf{g}_{s,\mathbf{h},\mathbf{t}}$, and the rotation required to go from **t** to **h** is denoted $\mathbf{g}_{s,\mathbf{t},\mathbf{h}}$.
In the case of antisymmetric relations, if $(\mathbf{h}, \mathbf{r}, \mathbf{t})$ is valid, then $(\mathbf{t}, \mathbf{r}, \mathbf{h})$ should not be valid unless the head and tail are the same entity. This is constrained in our model by not allowing a valid reverse triplet unless the two entities in question are equivalent.
We can define inversion relationships in our model by using a specific rotation operation. When this rotation is applied to the head entity, the result is that the tail entity becomes the inverse of what the head entity is. This is a way of making our model detect specific pairs of related entities that have an inverted relationship.
Composition of relationships can also be represented by RotatE. For instance, if we have two relationships r_1 and r_2, the model can learn a new rotation r_3 that represents the relationship formed by the composition of r_1 and r_2. This makes it possible to represent in the knowledge graph rich, multi-hop relationships in a way that is (a) plausible and (b) interpretable, since we can discern what types of relationships between entities lead to the formation of the represented relationship.
Taking advantage of the characteristics of complex rotations and the Hadamard product, RotatE presents a flexible and expressive framework for depicting and deducing associations in knowledge graphs. This makes RotatE a significant advancement in knowledge graph representation and reasoning over its predecessors.

IV. Alignment of Word Embeddings and Knowledge Graph Embeddings

A. Importance of Alignment

Aligning word embeddings with knowledge graph embeddings is a crucial step toward integrating the rich semantic information captured by language models with the structured, factual knowledge encoded in KGs. This alignment enables LLMs to take advantage of both the contextual understanding of language and the relational data of KGs, potentially reducing hallucinations and improving the trustworthiness of the generated output.

By mapping words and entities into a shared vector space where semantic similarities and relational structures are preserved, we achieve the following:

1) **Enhanced Semantic Understanding**: Combining contextual word meanings with factual entity relationships. Aligning word embeddings with KG embeddings has one primary benefit, and that is semantic understanding. Word embeddings capture meaning; that is, they allow us to understand the contextual meanings of words based on their use in a large corpus of text. But word embeddings don't just give us meaning; they also represent a kind of knowledge. When we talk about "embedding knowledge," that is essentially what we mean. Contemporary word embeddings allow us to understand not just the types of context in which a word appears but also the relationships between words, that is, the facts about the kinds of semantic entity that words represent.
2) **Improved Reasoning Capabilities**: The alignment of word and knowledge graphs also improves the reasoning capabilities of large language models. Knowledge graphs are structured representations that capture the complex relationships and hierarchies among entities. They enable the formation of logical inferences. For even simple tasks, such as answering questions about basic facts or making decisions, these structures add a lot and make the models usable in many more contexts.
3) **Intrinsic Believability Evaluation**: Allows LLMs to assess the plausibility of information without external references. When words and entities are mapped into a shared vector space, the LLM can evaluate the plausibility of the information based on a dual assessment of language and structured knowledge. It identifies inconsistencies and implausible-seeming statements without relying solely on external references or datasets. Consider the following statement: "Cats can fly." An LLM could evaluate that claim against its knowledge about cats and their biology to arrive at a judgment about the statement's validity that is much closer to true or false than anything a human annotator could approximate when grading the sentence.

This alignment is vital for improving the interoperability of textual and structured data, allowing much more effective information retrieval, question answering, and a host of other applications. Three main approaches can be used to achieve this alignment: mapping-based methods, joint embedding methods, and graph neural networks (GNNs).

B. Methods for Alignment

1. Mapping-Based Methods

Mapping-based methods involve learning a transformation function that maps embeddings from one vector space to another. Typically, a linear or non-linear mapping is learned using a set of aligned word-entity pairs.

1) **Linear Transformation** Mikolov *et al*. [20] proposed learning a linear mapping between monolingual word embeddings to perform cross-lingual word translation. The idea of linear mapping of linguistic units was introduced for the task of translating cross-lingual words. This method imposes a linear transformation on the word embeddings in such a way that the distance to the corresponding KG embeddings is minimized.
2) **Orthogonal Procrustes Analysis** The Orthogonal Procrustes problem seeks an optimal rotation matrix that aligns two sets of embeddings while preserving their geometric properties, ensuring that the mapping is distance-preserving and reduces distortion. Xing *et al*. [21] applied this technique to ensure that the transformation they produced from the original embeddings to the target embeddings was a distance-preserving mapping, which means that they were distorting the embeddings as little as possible and trying to maintain the integrity of the original "embedding space."
3) **Non-Linear Mapping** Non-linear functions, such as neural networks, can model more complex relationships between embedding spaces. Wang *et al*. [22] employed deep learning for cross-modal embedding alignment of words and entities, to achieve a more nuanced and complex representation that for more flexibility in representing intricate relationships between words and entities.

2. Joint Embedding Methods

Joint embedding methods aim to learn embeddings for words and KG entities within a shared vector space simultaneously. This approach promotes embeddings that reflect combined semantic and relational information by leveraging both textual and graph data during training.

1) **Knowledge-Enhanced Skip-Gram** Cui *et al*. [23] proposed incorporating KG information into the Skip-Gram model by adjusting the loss function to account for KG relations. This method builds on the traditional Skip-Gram model, which usually generates embeddings by concentrating on word cooccurrences. The Knowledge-Enhanced Skip-Gram model is an improvement over the basic Skip-Gram method; its loss function has been altered to include relations from a knowledge graph. When using this model, one generates embeddings that reflect much more than just the linguistic context of the words—they also reflect the structural relationships between the

entities in the knowledge graph. In this way, the Knowledge-Enhanced Skip-Gram model returns embeddings that are both richer and more useful. By integrating entity relationships directly into the training objective, the resulting embeddings capture both word cooccurrences and KG structures.

2) **Retrofitting** Faruqui *et al*. [24] introduced retrofitting, where pretrained word embeddings are adjusted post hoc to better fit a semantic lexicon or KG. This technique is based on the refinement of pre-trained word embeddings, such that they become more semantically aligned with a given lexicon or knowledge graph. The retrofitting process then involves a quasi-supervised adjustment of the embeddings, based on an objective function that nudges not-so-related words and entities to closely trudge in the same vector space. In other words, the objective function encourages embeddings of related words/entities to be close in the vector space, refining the embeddings to more closely reflect external knowledge sources.
3) **Joint Learning Models** Yao *et al*. [25] developed KG-BERT, a model that integrates KG information into BERT by jointly learning embeddings for both entities and relations during the language model training process. This joint learning allows KG-BERT to be much more factual in its understanding of the world, which is of utmost importance for tasks like knowledge graph completion, a task where essentially one must provide the missing links between already known entities and relations. The KG-BERT version of that task should, in principle, work much better than a BERT-only version.

3. Graph Neural Networks (GNNs)

Graph Neural Networks [26] leverage the structure of knowledge graphs to propagate information and learn embeddings that capture both local and global graph properties. KGs typically have intricate structures, making GNNs a natural solution to leveraging their intrinsic properties and using them to learn better embeddings.

1) **Relational Graph Convolutional Networks (R-GCNs)** Schlichtkrull *et al*. [27] extended Graph Convolutional Networks to handle multirelational data in KGs. R-GCNs aggregate information from neighboring nodes and relations, updating the embeddings of the entities based on the structure of the graph. Furthermore, R-GCNs work on not just single-relational graphs but also multi-relational graphs in something like a half-ASR mode. The models allow for multi-hop reasoning and fine-grained entity representation.
2) **Knowledge Graph Attention Networks (KGAT)** Wang *et al*. [28] proposed KGAT, which incorporates attention mechanisms that prioritize the importance of various neighboring nodes and relations during the embedding update process. With this attention-based approach, KGAT can focus on the most relevant and significant parts of a knowledge graph when learning representations. This dynamic focus improves the efficiency and effectiveness of KGAT compared to prior models. Finally, KGAT weighs the contributions of different nodes and relationships in a knowledge graph to better adapt to the local context of a graph when performing a specific task.
3) **Integration with Language Models** Lin *et al*. [29] introduced KAGNet, which integrates the GNN framework with a language model, thereby ensuring that both the "embeddings" derived from a knowledge graph and the structured reasoning capabilities of the GNN work harmoniously with the language model. By integrating KG-derived embeddings into the language model, the system benefits from both contextual language understanding and structured knowledge reasoning.

C. Challenges in Alignment

While embedding alignment holds significant promise, several challenges must be addressed:

1) **Semantic Heterogeneity** Semantic heterogeneity arises from differences in how concepts are represented in text and KGs. The main source of semantic heterogeneity is *polysemy*, in which a single word carries different meanings in different contexts. The term "bank," for example, can mean a financial institution or the shore of a river. This kind of ambiguity makes it difficult to align two knowledge graphs, since you have to know which structure is which to understand the whole picture. Speaking of whole pictures, KGs can also contain entities with ambiguous labels; that is, two or more entities that could share the same name. Ambiguous labels make mapping between KGs much harder.

   The alignment process is vulnerable to errors if it lacks strong disambiguation mechanisms. Incorrect associations between words and entities can spread throughout the system. These inaccuracies undermine not just the alignment's integrity, but also the trustworthiness of conclusions and insights said to be derived from the integrated data. It is absolutely essential to sort out any misunderstandings—semantic heterogeneities, as it were—before one can reasonably expect the embedding alignment to pay off. In our case, we discuss connecting natural language to knowledge graphs. Techniques such as entity linking and word sense disambiguation are

critical for accurate mapping. Without proper disambiguation, the alignment may associate words with incorrect entities, leading to the propagation of errors.

2) **Scalability** At scale, aligning embeddings poses computational challenges as a result of the large size of vocabularies and knowledge graphs. It is vital to create scalable algorithms capable of processing a large number of words and entities. Computational demand can be better managed by using methods like negative sampling, hierarchical softmax, and mini-batch training. They reduce the work necessary to update the model parameters during the training phase. Distributed computing and parallel processing are also valuable for handling large-scale data.
3) **Data Quality Issues** The effectiveness of the alignment process between word embeddings and knowledge graph embeddings is mainly governed by the quality of KG data. There are a number of challenges that come from the intrinsic characteristics of the data that can significantly affect the performance of the embedding method.
    1) **Incomplete or Noisy Data** Knowledge graphs often contain incomplete or noisy data. This stems from the processes used to gather the information, as well as from human errors during the annotation stage. These issues lead to gaps in the representation of both the entities and their relationships that affect the quality of the generated embeddings. Some embedding methods address this issue, but not all of them do. Embedding methods must be robust to such imperfections, possibly by incorporating uncertainty modeling or confidence scores for KG facts.
    2) **Bias in Knowledge Graphs** Biases present in KGs can be propagated to embeddings, affecting downstream applications. Awareness of potential biases and the implementation of mitigation strategies, such as bias correction techniques or fairness-based learning algorithms, are necessary to ensure ethical AI systems.
4) **Alignment Evaluation** Assessing the quality of alignment between word embeddings and KG embeddings is a challenge. Standardized metrics for evaluating alignment are lacking. The construction of solid assessment indexes is necessary to fill this gap. These indexes must incorporate various aspects including semantic similarity, relational accuracy, and impact on downstream tasks. The development of these metrics is crucial to benchmarking and improving alignment methods. Setting these metrics permits a more systematic evaluation of alignment methods and paves the way for improvements in the integration of word and Knowledge Graph embeddings.

Addressing these challenges is essential for the successful integration of word and KG embeddings, which in turn can significantly enhance the trustworthiness of LLMs by enabling intrinsic believability evaluation and reducing the occurrence of hallucinations. The methods discussed offer promising directions but require careful consideration of the associated complexities.

V. Survey of Current Methods

In this section, we survey contemporary methods that integrate word embeddings with knowledge graphs (KGs) to enhance large language models (LLMs). The focus is on techniques that aim to improve semantic understanding, reduce hallucinations, and enable evaluation of intrinsic believability in LLM.

A. Word Embedding Enhancement via Knowledge Graphs

Integrating knowledge graphs with word embeddings enriches semantic representations by leveraging structured knowledge and expanding context. Two primary approaches are:

1. Knowledge-Enhanced Contextual Word Representations

These methods integrate KG information during pre-training or fine-tuning of language models to enhance contextual word embeddings.

1) **Enhanced Representation through kNowledge IntEgration (ERNIE)** [30] incorporates entity representations from KGs into BERT-like architectures. It injects entity embeddings aligned with words in the input text, enabling the model to capture both contextual and factual knowledge. Consequently, the ERNIE model is able to take in not just the contextual language content of its input, but also the kind of knowledge that is associated with specific entities. It performs better than BERT on tasks that demand an understanding of both languages, such as question answering and entity recognition.
2) **K-BERT** [31] extends BERT by injecting KG triples into input sequences using a soft position and visible matrix to prevent information leakage. This allows to model to integrate structural knowledge from KGs into the

language model without significantly increasing computational complexity. When K-BERT does this, it still needs to process the knowledge graph information in a way that doesn't affect the input to the BERT model or compromise the model's performance. K-BERT does this by employing a visibility matrix and a position matrix.

3) **KnowBERT** [32] augments BERT with knowledge embeddings by integrating entity linker and KG embeddings into intermediate layers of the transformer model. This allows the model to incorporate the entity-level information from KGs directly into its contextual representations.
Imagine a human reader understanding a sentence involving an entity, like "Barack Obama," or a more obscure entity, like a "niche" plant or animal. The reader might mentally consult what they know (or could easily look up) about the entity to comprehend the sentence. KnowBERT's contextualized representations involve the same sort of feat at scale.

2. Retrofitting Word Embeddings

Retrofitting adjusts pre-trained word embeddings to better fit external semantic resources, such as KGs or lexical databases.

1) **Original Retrofitting** Faruqui *et al*. [24] introduced retrofitting by modifying word embeddings post-training to make them more compatible with semantic lexicons. The method minimizes the distance between embeddings of related words as defined by the external resource while preserving the original embedding space's structure. The external resource used in their experiments was the WordNet lexical database. It contains rich semantic information and is a good candidate for the kind of external resource that one would want to use for retrofitting.
2) **Attract-Repel Method** Mrkšić *et al*. [33] introduced the Attract-Repel method, which enhances the original retrofitting procedure by adding antonymy constraints. The method is based on the principle that synonyms should be attracted to each other in the embedding space, while antonyms should be repelled to opposite sides. By modeling these relationships, the Attract-Repel method refines the embeddings' semantic structure, yielding a more fine-grained representation of language. This approach not only makes the embeddings better at capturing synonymy but also gives them a much stronger ability to distinguish between words with opposite meanings.
3) **Graph-Based Retrofitting** Hamilton *et al*. [34] developed a graph-based retrofitting approach that uses the structure of KGs to adjust embeddings. The method propagates embedding adjustments through the graph, ensuring consistency across related entities and relations. In these graphs, entities (in our case, words) are represented as nodes, with the relationships between them forming the edges. When a knowledge graph is used to inform the retrofitting process, it provides a clear picture of where and how to adjust the embeddings.

B. Knowledge Graph Embedding Techniques Relevant to LLMs

The embedding methods used for Knowledge Graphs are essential to merge structured knowledge into Large Language Models. They not only enrich the knowledge and structure of the models, but also offer a clear pathway to "ground" them and make sense of the information they contain. Fortunately, a great number of tools and techniques have been developed for performing these embeddings. The basic components of these embeddings are entity recognition and entity linking.

1. Entity Linking and Recognition

Entity Linking involves connecting the mentions found in texts with their respective entities in a Knowledge Graph. The association that this linking establishes enables the models to use knowledge in a more structured form, which directly benefits the models' comprehension skills and, by extension, the contextual awareness they exhibit.

1) **Neural Entity Linking** Kolitsas *et al*. [35] proposed an end-to-end neural model for entity linking. The model combines contextual word embeddings with entity embeddings derived from knowledge graphs in a way that enhances the accuracy of both tasks that make up entity linking: recognition and disambiguation. In fact, the model integrates two types of embedding that are increasingly being viewed as necessary to achieve state-of-the-art results in these tasks. Contextual word embeddings hold the nuances of language and the specific context in which an entity is mentioned; and entity embeddings provide a structured representation of the entities themselves. Disambiguation can happen more effectively when you have these two types of embeddings on hand.
2) **Zero-Shot Entity Linking** Logeswaran *et al*. [36] introduced a zero-shot entity linking model that generalizes to unseen entities by representing them through descriptions and KG embeddings. This method uses descriptions and KG embeddings to characterize unseen entities, enabling linking with high confidence and minimal prior

knowledge. The approach, therefore, has substantial advantages when new entities appear frequently, as in many contemporary contexts. By not depending on detailed prior knowledge, the work makes substantial strides in entity linking.

2. Relation Extraction

Relation extraction involves methods which determine the relationships (if any) between the entities that are mentioned in a given piece of text, facilitating the integration of new information into KGs.

1) **Distant Supervision** Mintz *et al*. [37] introduced a technique that takes advantage of the KGs that are currently available to create training data for relation extraction models. KGs are used to connect entities that have relationships within the KG. Any sentence that mentions a pair of entities that are connected in the KG can be viewed as a valid relation instance. In this way, one can use the KGs to separate all the sentences that can be used to train a model to extract relations. This method has a very broad coverage and is, therefore, useful to create an initial set of relations. However, it suffers from noise and can be readily improved.
2) **Neural Relation Extraction** Zeng *et al*. [38] developed a convolutional neural network model for relation extraction that captures lexical and sentence-level features. By incorporating KG embeddings, the model can improve its ability to identify and classify relations in text.
3) **Joint Learning Approaches** Ji *et al*. [39] proposed a joint learning framework that simultaneously performs entity recognition, entity linking, and relation extraction. This integrated approach allows for improved representation sharing across tasks and, hence, improved accuracy in structuring text to align with knowledge graphs.
   Recognizing and linking entities concurrently with relation extraction also allows for much more effective leveraging of contextual information, particularly when entity types are involved, and leads to a much more coherent understanding of the text's portrayal of relationships. By integrating these tasks, the model benefits from shared representations and can better align text with KG structures.

C. Trust and Belief Metrics in Knowledge Graphs

Incorporating trust and belief metrics into KGs enhances the ability of LLMs to evaluate the plausibility of information.

1. Belief Propagation Models

Belief propagation models assign confidence scores to entities and relations, enabling the assessment of information reliability.

1) **Probabilistic Soft Logic (PSL)** Bach *et al*. [40] introduced PSL, a framework for probabilistic reasoning over relational data. It allows the incorporation of uncertainty and belief propagation in KGs by formulating logical rules with associated confidence levels. Using this framework, one can assign confidence levels to the rules, with higher confidence levels indicating greater certainty that the rule is true, and lower confidence levels indicating the opposite.
2) **Trust Propagation** Ziegler and Lausen [41] proposed algorithms for propagating trust in social networks, which can be adapted for KGs. By modeling trust as a transitive property, the confidence in certain facts can be inferred based on the trustworthiness of connected entities and relations. If entity A is trustworthy and entity B is directly connected to entity A, then entity B can be inferred to be trustworthy as well (unless, of course, there's some reason to conclude otherwise). A "trustworthy" entity is, in effect, a "credible" source of information.

2. Uncertainty Modeling

Modeling uncertainty helps handle ambiguous or conflicting information in KGs.

1) **Fuzzy Logic in KGs** Parvizi *et al*. [42] applied fuzzy logic to KGs to represent uncertain knowledge. This approach assigns degrees of truth to facts, allowing the KG to capture nuances in belief.
2) **Bayesian Networks** Friedman *et al*. [43] used Bayesian networks to model probabilistic relationships between entities and attributes in KGs. This method enables reasoning under uncertainty and can update beliefs based on new evidence.

The surveyed methods collectively illustrate the significant advancements made in integrating knowledge graphs with large language models. By combining the strengths of contextual language understanding and structured knowledge representation, these approaches contribute to the development of more trustworthy and capable AI systems. They

address key challenges such as hallucinations and misinformation by providing mechanisms for LLMs to access, reason about, and generate information grounded in factual data.

VI. Applications Leading to Trust-Informed LLMs

A. Intrinsic Believability Evaluation Mechanisms

Intrinsic believability evaluation enables LLMs to assess the plausibility of information without relying on external verification systems. By integrating KGs and embedding alignment, LLMs can internally verify facts and reason about the truthfulness of generated content.

1. Knowledge-Augmented Language Models

1) **Fact-Enhanced Language Models** Logan *et al*. [44] introduced a fact-enhanced language model that incorporates factual knowledge during text generation. By conditioning the language model on KG-derived facts, the model can generate text that is both contextually relevant and factually accurate.
2) **Commonsense Reasoning Models** Bosselut *et al*. [45] developed COMET, a model that generates commonsense knowledge by integrating KGs like ConceptNet into the language modeling process. This integration allows the model to produce more plausible and coherent outputs by leveraging commonsense reasoning.
3) **Pre-trained Models with Integrated KGs** Wang *et al*. [46] proposed KEPLER, a model that jointly learns textual and KG embeddings during pre-training. By aligning entity representations in text with those in the KG, the model enhances its ability to understand and generate factually consistent information.

2. Hallucination Reduction Techniques

Reducing hallucinations involves constraining the language model to generate outputs grounded in factual knowledge.

1) **Controlled Generation with KGs** Yu *et al*. [47] introduced methods for controlled text generation using KGs to guide the content and style of the output. By incorporating KG constraints, the model avoids generating content that deviates from known facts.
2) **Verification Modules** Kadavath *et al*. [48] proposed integrating verification modules into LLMs that cross-check generated content against KGs. This process involves querying the KG during generation to confirm the validity of statements, thereby reducing hallucinations.

B. Reinforcing LLMs with Knowledge Graphs

Reinforcing LLMs with KGs enhances their reasoning capabilities and factual consistency by incorporating structured knowledge into the generation process.

1. Structural Reinforcement

1) **Graph-Structured Representations** Cai and Lam [49] proposed models that utilize graph-structured representations of text and knowledge to improve language understanding. By encoding both the syntactic structure of sentences and the relational structure of KGs, the model enhances its comprehension and generation abilities.
2) **Memory-Augmented Neural Networks** Kumar *et al*. [50] developed Dynamic Memory Networks that incorporate external memory components storing KG information. This allows the model to access and manipulate structured knowledge during reasoning tasks.

2. Semantic Reinforcement

1) **Semantic Supervision** Zhang *et al*. [51] introduced semantic supervision by aligning language model outputs with KG semantics. By training the model to generate text that is semantically consistent with KG relations, it learns to produce more accurate and meaningful content.
2) **Knowledge-Aware Attention Mechanisms** Liu *et al*. [52] incorporated knowledge-aware attention mechanisms into sequence-to-sequence models. By attending to relevant KG entities and relations during generation, the model ensures that outputs are grounded in factual knowledge.

C. Maintaining Belief Metrics in Language Models

Incorporating belief metrics into LLMs allows for the evaluation and adjustment of confidence levels in generated content based on KG information.

1. Edge Weighting Techniques

1) **Confidence-Weighted Decoding** Holtzman *et al*. [53] proposed confidence-weighted decoding strategies where the language model's predictions are adjusted based on the confidence levels derived from KGs. This method reduces the probability of generating low-confidence or incorrect statements.
2) **Probabilistic Graphical Models** He *et al*. [54] integrated probabilistic graphical models with LLMs to represent uncertainty and belief in generated content. By modeling the dependencies between variables and incorporating KG-derived probabilities, the model can produce outputs with calibrated confidence levels.

2. Updating Belief Metrics

1) **Dynamic Knowledge Integration** Wang *et al*. [55] introduced dynamic knowledge integration methods where the language model updates its internal belief states based on new information from KGs. This enables the model to adapt to changes in knowledge and maintain up-to-date representations.
2) **User Feedback Incorporation** Li *et al*. [56] proposed incorporating user feedback into LLMs to adjust belief metrics. By allowing users to provide corrections or confirmations, the model refines its beliefs and improves future outputs.

D. Domain-Specific Applications

Applying these methods to domain-specific contexts demonstrates their effectiveness in enhancing trustworthiness in specialized areas.

1. Healthcare Applications

1) **Clinical Decision Support** Shao *et al*. [57] integrated medical knowledge graphs with LLMs to assist in clinical decision-making. By grounding language models in validated medical knowledge, the system provides accurate and reliable recommendations.
2) **Biomedical Text Generation** Peng *et al*. [58] explored biomedical text generation using KGs to ensure that generated content adheres to medical facts and terminology. This reduces the risk of misinformation in critical healthcare communications.

2. Legal and Financial Applications

1) **Legal Document Analysis** Zhong *et al*. [59] incorporated legal knowledge graphs into language models for analyzing legal documents. By leveraging structured legal knowledge, the model enhances its understanding of legal concepts and provides more accurate analyses.
2) **Financial Reporting** Yang *et al*. [60] developed FinBERT, a language model tailored for financial text analysis. Integrating financial KGs helps the model interpret complex financial terminology and generate reliable insights.

E. Real-Time Systems and Scalability

Addressing scalability and real-time processing is essential for deploying trust-informed LLMs in practical applications.

1. Efficient Integration Techniques

1) **Sparse Attention Mechanisms** Child *et al*. [61] proposed sparse attention mechanisms to reduce computational overhead in transformer models. Efficient attention enables the integration of large KGs without prohibitive computational costs.
2) **Knowledge Distillation** Sanh *et al*. [62] utilized knowledge distillation to create smaller, efficient models that retain the capabilities of larger models. By distilling KG-enhanced LLMs into compact versions, deployment in resource-constrained environments becomes feasible.

2. Real-Time Knowledge Updates

1) **Incremental Learning** Parisi *et al*. [63] explored incremental learning techniques for LLMs, allowing models to update their knowledge bases in real-time as new data becomes available. This ensures that the model's outputs remain current and accurate.
2) **Streaming Knowledge Integration** Jin *et al*. [64] developed methods for integrating streaming knowledge into LLMs. By processing KG updates as streams, the model maintains up-to-date information without the need for complete retraining.

Collectively, these applications illustrate the significant advancements in developing trust-informed LLMs through the integration of knowledge graphs. By enabling models to internally evaluate the believability of information and grounding their outputs in structured knowledge, we move closer to addressing critical issues such as hallucinations and misinformation in AI-generated content.

VII. Challenges and Open Issues

Despite the significant advancements in integrating knowledge graphs with large language models, several challenges and open issues remain. Addressing these challenges is crucial for the continued development of trustworthy and reliable AI systems capable of intrinsic believability evaluation.

A. Technical Challenges

1. Integration Complexity

1) **Model Architecture Compatibility** Integrating KGs with LLMs often requires modifications to the model architecture, which can introduce complexity and affect the stability of the models. Ensuring compatibility between different embedding spaces and architectures is non-trivial. Techniques like fine-tuning and adapter modules can help, but may not fully resolve compatibility issues.
2) **Computational Overhead** The incorporation of KGs and embedding alignment increases computational requirements for both training and inference. Models become larger and more resource-intensive, posing challenges for deployment, especially in real-time systems or resource-constrained environments. Efficient algorithms and optimization strategies are needed to mitigate computational overhead.

2. Scalability

1) **Handling Large-Scale Knowledge Graphs** Knowledge graphs can be massive, containing millions of entities and relationships. Scaling embedding alignment and integration methods to handle such large graphs is challenging. Memory constraints and computational efficiency must be addressed to process and utilize KGs effectively.
2) **Real-Time Processing** Real-time applications require prompt responses, but integrating KGs can introduce latency due to additional computations. Optimizing models for speed without sacrificing accuracy is essential for practical deployment in time-sensitive applications.

B. Data-Related Challenges

1. Knowledge Graph Quality

1) **Incomplete and Noisy Data** Knowledge graphs may contain incomplete information, errors, or outdated data. Relying on such KGs can lead to incorrect inferences and reduce the trustworthiness of LLM outputs. Developing methods to assess and improve KG quality, such as data cleansing and validation techniques, is critical.
2) **Bias in Knowledge Graphs** KGs may reflect biases present in their source data, leading to biased or unfair model outputs. For example, overrepresentation or underrepresentation of certain groups can result in discriminatory behavior by the AI system. Identifying and mitigating biases in KGs is necessary to ensure ethical AI practices.

2. Semantic Alignment

1) **Entity and Relation Mapping** Aligning entities and relations between text and KGs is challenging due to differences in terminology, language ambiguity, and polysemy. Effective entity linking and relation extraction

methods are required to accurately map textual information to KG concepts.

2) **Cross-Lingual and Domain-Specific Variations** Language models may struggle with cross-lingual alignment or domain-specific terminology not well-represented in general-purpose KGs. Developing specialized KGs and models capable of handling multiple languages and domains is an open area of research.

C. Methodological Challenges

1. Evaluation Metrics

1) **Assessing Alignment Quality** Standardized metrics for evaluating the quality of embedding alignment and KG integration are lacking. Developing comprehensive evaluation frameworks that measure semantic similarity, factual accuracy, and impact on downstream tasks is necessary for benchmarking methods and guiding improvements.
2) **Measuring Trustworthiness** Quantifying the trustworthiness and believability of LLM outputs is challenging. Metrics that capture the correctness, reliability, and confidence of generated content need to be established to assess the effectiveness of trust-informed models.

2. Model Interpretability

1) **Understanding Decision Processes** As models become more complex with KG integration, interpreting how they arrive at specific outputs becomes more difficult. Enhancing model interpretability is important for debugging, trust, and compliance with transparency requirements.
2) **Explainable AI Techniques** Incorporating explainable AI (XAI) techniques can help users understand the reasoning behind model outputs. Developing XAI methods tailored for KG-enhanced LLMs is an ongoing challenge that requires further research.

D. Ethical and Social Considerations

1. Privacy Concerns

1) **Sensitive Information in Knowledge Graphs** KGs may contain sensitive or personal data, raising privacy concerns when integrated with LLMs. Ensuring compliance with data protection regulations, such as GDPR, and implementing privacy-preserving techniques are essential.
2) **Data Anonymization** Techniques for anonymizing data within KGs without losing critical information are needed. Balancing the utility of the KG with privacy protection remains a significant challenge.

2. Fairness and Bias Mitigation

1) **Algorithmic Fairness** Ensuring that LLMs do not produce biased or discriminatory outputs is crucial. Biases in KGs and training data can propagate through the model. Developing fairness-aware algorithms and regularization techniques can help mitigate these issues.
2) **Ethical AI Guidelines** Adhering to ethical AI guidelines and frameworks is important for responsible deployment. Incorporating ethical considerations into model development and evaluation processes is necessary to build trust with users and stakeholders.

3. Misuse of Technology

1) **Misinformation and Disinformation** While integrating KGs can enhance trustworthiness, there is a risk that advanced models could be misused to generate convincing but false information. Implementing safeguards and monitoring mechanisms is important to prevent the dissemination of disinformation.
2) **Dual-Use Concerns** AI technologies can be used for both beneficial and harmful purposes. Awareness of dual-use implications and establishing policies to restrict malicious applications are essential for ethical AI practice.

E. Open Issues

1. Dynamic Knowledge Integration

Developing methods for continuous and real-time integration of new knowledge into LLMs remains an open issue. Models need to adapt to new information without requiring extensive retraining.

2. Standardization and Benchmarking

The lack of standardized datasets and benchmarks for evaluating KG-enhanced LLMs hinders progress. Creating shared resources and evaluation protocols would facilitate comparison and accelerate advancements in the field.

3. Generalization Across Domains

Ensuring that methods for embedding alignment and KG integration generalize well across different domains and languages is a challenge. Research is needed to develop models that are robust and adaptable to various contexts.

F. Potential Solutions and Research Directions

1) **Advancements in Model Architectures** Exploring new architectures that inherently support KG integration, such as hybrid models combining symbolic and neural approaches, may address compatibility and scalability issues.
2) **Efficient Algorithms and Optimization** Developing more efficient algorithms for embedding alignment and KG integration can reduce computational overhead. Techniques like sparse representations, quantization, and model pruning may help.
3) **Improved Data Practices** Enhancing KG quality through better data collection, cleansing, and validation practices can mitigate issues related to incomplete and noisy data. Collaborative efforts to build high-quality, bias-aware KGs are valuable.
4) **Interdisciplinary Collaboration** Collaboration between researchers in NLP, knowledge representation, ethics, and social sciences can lead to more holistic solutions addressing both technical and ethical challenges.
5) **Community Standards** Establishing community standards for evaluation, data sharing, and ethical practices can facilitate progress and ensure responsible development of trust-informed LLMs.

Advancements in model architectures that inherently support KG integration may address compatibility and scalability issues. Developing more efficient algorithms and optimization techniques can reduce computational overhead. Improving data practices through better collection, cleansing, and validation can enhance KG quality. Interdisciplinary collaboration and the establishment of community standards for evaluation and ethical practices are also crucial for fostering progress in this field.

In summary, while significant progress has been made in integrating KGs with LLMs to enhance trustworthiness, addressing the challenges and open issues identified is essential to realize the full potential of trust-informed AI systems. Continued research and collaboration are needed to develop models that are not only powerful but also reliable, interpretable, and aligned with ethical standards. The next section will explore **future directions** that build on these insights, pointing toward promising avenues for advancing the state-of-the-art in this domain.

VIII. Conclusion

In this paper, we have conducted a comprehensive survey of the integration of word embeddings and knowledge graphs to enhance the trustworthiness and reasoning capabilities of large language models. The rapid advancement of LLMs has led to remarkable achievements in natural language understanding and generation. However, challenges such as hallucinations, lack of interpretability, and the propagation of biases have raised concerns about their reliability and trustworthiness. Addressing these issues is critical as AI systems become increasingly integrated into various aspects of society.

We began by exploring the **fundamentals** of word embeddings and knowledge graphs, establishing the foundational concepts necessary for understanding their alignment. Traditional word embeddings like *Word2Vec*, *GloVe*, and *FastText* capture semantic relationships based on co-occurrence statistics, providing static representations of words. Contextualized embeddings, introduced by models like *ELMo*, *BERT*, and *GPT*, generate dynamic representations that consider the context in which words appear, leading to significant improvements in various NLP tasks. Knowledge

graphs, on the other hand, offer structured representations of entities and their relationships, enabling machines to reason over factual information.

The **alignment** of word embeddings with KG embeddings bridges the gap between unstructured textual data and structured knowledge bases. By mapping words and entities into a shared vector space, we enable LLMs to leverage both the rich contextual semantics of language models and the precise, factual information contained in KGs. This integration allows for improved reasoning capabilities, enhanced semantic understanding, and the ability to perform intrinsic believability evaluation—assessing the plausibility of information without external verification.

Our **survey of current methods** revealed a diverse range of techniques for embedding alignment. Mapping-based methods learn transformations between embedding spaces, enabling cross-modal and cross-lingual applications. Joint embedding approaches simultaneously learn representations for words and KG entities, fostering embeddings that capture both linguistic context and factual relationships. Graph neural networks exploit the structural properties of KGs to learn embeddings that incorporate both local and global graph information. These methods have been successfully applied in various domains, including open-domain question answering, dialogue systems, and information verification, demonstrating their effectiveness in enhancing LLM performance and reducing hallucinations.

Despite these advancements, we identified several **challenges and open issues**. Technical challenges such as integration complexity and computational overhead hinder the seamless incorporation of KGs into LLMs. Data-related issues, including the quality, completeness, and bias present in KGs, affect the reliability of AI outputs. Methodological challenges, such as the lack of standardized evaluation metrics and the difficulty in interpreting complex models, impede progress and adoption. Ethical and social considerations, including privacy concerns, fairness, and the potential misuse of technology, underscore the importance of responsible AI development.

It is important to recognize that while **hallucination mitigation strategies** aim to enhance the reliability of LLM outputs, they may inadvertently **hamper the development of novel logical reasoning abilities** in future models. By constraining models to strictly adhere to existing knowledge bases and reducing their propensity to generate imaginative or speculative content, we risk limiting their capacity for creative problem-solving and innovative reasoning. The balance between preventing misinformation and fostering the emergence of advanced reasoning capabilities is delicate. Overemphasis on hallucination reduction may stifle the model's ability to make novel inferences, explore hypothetical scenarios, or generate original ideas that extend beyond the current scope of available data.

Therefore, it is crucial to consider strategies that mitigate hallucinations without overly restricting the model's generative potential. Techniques that encourage controlled creativity, such as conditional generation with awareness of factual consistency, can help maintain a model's ability to engage in complex reasoning while ensuring outputs remain plausible and grounded in reality. Future research should focus on developing methods that allow models to **explore and reason beyond existing knowledge** in a responsible manner, fostering innovation without compromising trustworthiness.

In conclusion, the alignment of word embeddings and knowledge graph embeddings presents a significant opportunity to develop LLMs that are both trustworthy and capable of advanced reasoning. By integrating the rich semantic information from language models with the structured, factual knowledge of KGs, we can create AI systems that are more accurate, interpretable, and aligned with human values. Balancing hallucination mitigation with the encouragement of novel reasoning abilities is essential for advancing AI capabilities. **Continued research and interdisciplinary collaboration** are vital for overcoming the challenges identified and realizing the full potential of trust-informed AI. The future of AI hinges on our ability to build models that not only perform well but also innovate responsibly and earn the trust of the users they serve.

References

[1] Vylomova, E., Rimell, L., Cohn, T., and Baldwin, T., "Take and Took, Gaggle and Goose, Book and Read: Evaluating the Utility of Vector Differences for Lexical Relation Learning," *arXiv e-prints*, 2015, arXiv:1509.01692. https://doi.org/10.48550/arXiv.1509.01692.

[2] Mikolov, T., Chen, K., Corrado, G., and Dean, J., "Efficient Estimation of Word Representations in Vector Space," *Proceedings of the International Conference on Learning Representations (ICLR)*, 2013.

[3] Pennington, J., Socher, R., and Manning, C. D., "GloVe: Global Vectors for Word Representation," *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Association for Computational Linguistics, 2014, pp. 1532–1543.

[4] Bojanowski, P., Grave, E., Joulin, A., and Mikolov, T., "Enriching Word Vectors with Subword Information," *Transactions of the Association for Computational Linguistics*, Vol. 5, 2017, pp. 135–146.

[5] Peters, M. E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., and Zettlemoyer, L., "Deep Contextualized Word Representations," *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Association for Computational Linguistics, 2018, pp. 2227–2237.

[6] Radford, A., Narasimhan, K., Salimans, T., and Sutskever, I., "Improving Language Understanding by Generative Pre-Training," *OpenAI*, 2018.

[7] Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., and Sutskever, I., "Language Models are Unsupervised Multitask Learners," *OpenAI blog*, Vol. 1, No. 8, 2019, p. 9.

[8] Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., et al., "Language Models are Few-Shot Learners," *Advances in Neural Information Processing Systems*, Vol. 33, 2020, pp. 1877–1901.

[9] Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K., "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (NAACL-HLT)*, Association for Computational Linguistics, 2019, pp. 4171–4186.

[10] Vrandečić, D., and Krötzsch, M., "Wikidata: A Free Collaborative Knowledgebase," *Communications of the ACM*, Vol. 57, No. 10, 2014, pp. 78–85.

[11] Lehmann, J., Isele, R., Jakob, M., Jentzsch, A., Kontokostas, D., Mendes, P. N., et al., "DBpedia–A Large-Scale, Multilingual Knowledge Base Extracted from Wikipedia," *Semantic Web*, Vol. 6, No. 2, 2015, pp. 167–195.

[12] Bollacker, K., Evans, C., Paritosh, P., Sturge, T., and Taylor, J., "Freebase: A Collaboratively Created Graph Database for Structuring Human Knowledge," *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data*, ACM, 2008, pp. 1247–1250.

[13] Singhal, A., "Introducing the knowledge graph: Things, not strings," , May 2012. URL https://blog.google/products/search/introducing-knowledge-graph-things-not/.

[14] Bordes, A., Usunier, N., Garcia-Duran, A., Weston, J., and Yakhnenko, O., "Translating Embeddings for Modeling Multi-relational Data," *Advances in Neural Information Processing Systems 26 (NIPS)*, 2013, pp. 2787–2795.

[15] Wang, Z., Zhang, J., Feng, J., and Chen, Z., "Knowledge Graph Embedding by Translating on Hyperplanes," *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence*, AAAI Press, 2014, pp. 1112–1119.

[16] Lin, Y., Liu, Z., Sun, M., Liu, Y., and Zhu, X., "Learning Entity and Relation Embeddings for Knowledge Graph Completion," *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, AAAI Press, 2015, pp. 2181–2187.

[17] Yang, B., Yih, W.-t., He, X., Gao, J., and Deng, L., "Embedding Entities and Relations for Learning and Inference in Knowledge Bases," *Proceedings of the International Conference on Learning Representations (ICLR)*, 2015.

[18] Trouillon, T., Welbl, J., Riedel, S., Gaussier, É., and Bouchard, G., "Complex Embeddings for Simple Link Prediction," *Proceedings of the 33rd International Conference on Machine Learning (ICML)*, PMLR, 2016, pp. 2071–2080.

[19] Sun, Z., Deng, Z.-H., Nie, J.-Y., and Tang, J., "RotatE: Knowledge Graph Embedding by Relational Rotation in Complex Space," *International Conference on Learning Representations (ICLR)*, 2019.

[20] Mikolov, T., Le, Q. V., and Sutskever, I., "Exploiting Similarities among Languages for Machine Translation," *arXiv preprint arXiv:1309.4168*, 2013.

[21] Xing, C., Wang, J., Liu, K.-F., and Lin, W., "Normalized Word Embedding and Orthogonal Transform for Bilingual Word Translation," *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2015, pp. 1006–1011.

[22] Wang, Y.-N., and Li, Z., "Structural Similarities for Cross-Lingual Word Embeddings," *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, 2016, pp. 710–715.

[23] Cui, P., Wang, X., Pei, J., Zhu, W., and Zhang, S., "KD-LDAA: A Knowledge-Driven Approach to Learning Document and Author Embeddings," *2015 IEEE International Conference on Data Mining*, IEEE, 2015, pp. 271–280.

[24] Faruqui, M., Tsvetkov, Y., Rastogi, P., and Dyer, C., "Retrofitting Word Vectors to Semantic Lexicons," *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2015, pp. 1606–1615.

[25] Yao, S., Mao, C., and Luo, Y., "KG-BERT: BERT for Knowledge Graph Completion," *arXiv preprint arXiv:1909.03193*, 2019.

[26] Scarselli, F., Gori, M., Tsoi, A. C., Hagenbuchner, M., and Monfardini, G., "The Graph Neural Network Model," *IEEE Transactions on Neural Networks*, Vol. 20, No. 1, 2009, pp. 61–80. https://doi.org/10.1109/TNN.2008.2005605.

[27] Schlichtkrull, M., Kipf, T. N., Bloem, P., Van Den Berg, R., Titov, I., and Welling, M., "Modeling Relational Data with Graph Convolutional Networks," *European Semantic Web Conference*, Springer, 2018, pp. 593–607.

[28] Wang, X., He, X., Cao, Y., Liu, M., and Chua, T.-S., "KGAT: Knowledge Graph Attention Network for Recommendation," *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2019, pp. 950–958.

[29] Lin, B. Y., Zhou, W., Shen, M., Wang, P., and Ren, X., "KAGNet: Knowledge-Aware Graph Networks for Commonsense Reasoning," *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 34, 2020, pp. 8562–8569.

[30] Zhang, Z., Han, X., Liu, Z., Jiang, X., Sun, M., and Liu, Q., "ERNIE: Enhanced Language Representation with Informative Entities," *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 2019, pp. 1441–1451.

[31] Liu, W., Zhou, P., Zhao, Z., Wang, Z., Ju, Q., Deng, H.-G., and Wang, P., "K-BERT: Enabling Language Representation with Knowledge Graph," *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 34, No. 03, 2020, pp. 2901–2908.

[32] Peters, M. E., Neumann, M., Logan IV, R., Schwartz, R., Joshi, V., Singh, S., and Smith, N. A., "Knowledge Enhanced Contextual Word Representations," *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*, 2019, pp. 43–54.

[33] Mrkšić, N., Vulić, I., Ó Séaghdha, D., Leviant, I., Reichart, R., Gasperin, C., and Korhonen, A., "Semantic Specialisation of Distributional Word Vector Spaces using Monolingual and Cross-Lingual Constraints," *Transactions of the Association for Computational Linguistics*, Vol. 5, 2017, pp. 309–324.

[34] Hamilton, W. L., Leskovec, J., and Jurafsky, D., "Diachronic Word Embeddings Reveal Statistical Laws of Semantic Change," *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, 2016, pp. 1489–1501.

[35] Kolitsas, N., Ganea, O.-E., and Hofmann, T., "End-to-End Neural Entity Linking," *Proceedings of the 22nd Conference on Computational Natural Language Learning*, 2018, pp. 519–529.

[36] Logeswaran, L., Lee, K., Toutanova, K., Devlin, J., Lee, H., and Gardner, M., "Zero-Shot Entity Linking by Reading Entity Descriptions," *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 2019, pp. 3449–3460.

[37] Mintz, M., Bills, S., Snow, R., and Jurafsky, D., "Distant Supervision for Relation Extraction without Labeled Data," *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL*, 2009, pp. 1003–1011.

[38] Zeng, D., Liu, K., Lai, S., Zhou, G., and Zhao, J., "Relation Classification via Convolutional Deep Neural Network," *Proceedings of COLING 2014*, 2014, pp. 2335–2344.

[39] Ji, G., Liu, K., He, S., and Zhao, J., "Distant Supervision for Relation Extraction with Sentence-Level Attention and Entity Descriptions," *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 31, 2017.

[40] Bach, S. H., Broecheler, M., Huang, B., and Getoor, L., "Hinge-Loss Markov Random Fields and Probabilistic Soft Logic," *Journal of Machine Learning Research*, Vol. 18, No. 109, 2017, pp. 1–67.

[41] Ziegler, C.-N., and Lausen, G., "Propagation Models for Trust and Distrust in Social Networks," *Information Processing and Management of Uncertainty in Knowledge-Based Systems*, 2005, pp. 190–197.

[42] Parvizi, A., Gao, K., and Hitzler, P., "Fuzzy Knowledge Representation and Reasoning for Linked Open Data," *Proceedings of the International Conference on Web Intelligence*, 2015, pp. 57–64.

[43] Friedman, N., Getoor, L., Koller, D., and Pfeffer, A., "Learning Probabilistic Relational Models," *International Joint Conference on Artificial Intelligence*, Vol. 16, No. 2, 1999, pp. 1300–1309.

[44] Logan, R. L., Liu, N. F., Peters, M. E., and Gardner, M., "When Knowledge Fails: Deep Neural Networks Do Not Recognize Ungrammatical Sentences," *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*, 2019, pp. 16–20.

[45] Bosselut, A., Rashkin, H., Sap, M., Malaviya, C., Celikyilmaz, A., and Choi, Y., "COMET: Commonsense Transformers for Automatic Knowledge Graph Construction," *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 2019, pp. 4762–4779.

[46] Wang, Y., Gao, T., Zhu, Z., Liu, Z., Sun, M., and Zhou, J., "KEPLER: A Unified Model for Knowledge Embedding and Pre-trained Language Representation," *Transactions of the Association for Computational Linguistics*, Vol. 9, 2021, pp. 176–194.

[47] Yu, D., Hsu, C., Zhao, T., King, E., and Black, A. W., "Towards Controlling the Specificity of Dialogue Generation," *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 2020, pp. 3606–3613.

[48] Kadavath, S., et al., "Language Models Can Teach Themselves to Reason Better," *arXiv preprint arXiv:2205.11916*, 2022.

[49] Cai, D., and Lam, W., "Graph Transformer for Graph-to-Sequence Learning," *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 34, 2020, pp. 7464–7471.

[50] Kumar, A., Irsoy, O., Su, J., Bradbury, J., English, R., Pierce, B., Ondruska, P., Gulrajani, I., and Socher, R., "Ask Me Anything: Dynamic Memory Networks for Natural Language Processing," *Proceedings of the 33rd International Conference on Machine Learning*, 2016, pp. 1378–1387.

[51] Zhang, Y.-C., Sun, S.-A., Galley, M., Chen, Y.-C., Brockett, C., Gao, X., Gao, J., Liu, J., and Dolan, B., "Grounded Conversation Generation as Guided Traverses in Commonsense Knowledge Graphs," *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics*, 2019, pp. 2039–2049.

[52] Liu, P., Zhang, Y., Fu, J., Sugiyama, K., and Menzies, T., "Knowledge-Augmented Language Model and Its Application to Unsupervised Named-Entity Recognition," *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics*, 2018, pp. 1142–1150.

[53] Holtzman, A., Buys, J., Du, L., Forbes, M., and Choi, Y., "Surface Form Competition: Why the Highest Probability Answer Isn't Always Right," *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, 2021, pp. 7038–7051.

[54] He, R., and McAuley, J., "An Unsupervised Neural Attention Model for Aspect Extraction," *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, 2017, pp. 388–397.

[55] Wang, S., Zhang, Y., Sun, Q., Chen, Z., and Liu, T., "Dynamic Knowledge Graph Representation Learning for Knowledge-Driven Dialogue Generation," *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 2020, pp. 3307–3313.

[56] Li, J. J., Zhang, X. L., and He, J., "User Feedback in Natural Language Generation: A Survey," *Proceedings of the 14th International Conference on Natural Language Generation*, 2021, pp. 489–499.

[57] Shao, Y., Liang, Y., Wang, H., Zhang, Q., and Jiang, X., "Clinical Knowledge Graph Embedding for Healthcare," *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2021, pp. 2378–2382.

[58] Peng, W.-H., Wei, J., Lu, X., Kan, M.-Y., and Lin, S.-D., "An Empirical Study of Pre-trained Language Model for Biomedical Question Answering," *Proceedings of the 19th SIGBioMed Workshop on Biomedical Language Processing*, 2020, pp. 1–10.

[59] Zhong, H., Zhang, C., Tu, C., Xiao, C., Liu, Z., and Sun, M., "Iteratively Questioning and Answering for Interpretable Legal Judgment Prediction," *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 2020, pp. 3340–3353.

[60] Yang, H., Liu, Q., Li, J., and Yang, Z., "FinBERT: A Pretrained Language Model for Financial Communications," *arXiv preprint arXiv:2006.08097*, 2020.

[61] Child, R., Gray, S., Radford, A., and Sutskever, I., "Generating Long Sequences with Sparse Transformers," *arXiv preprint arXiv:1904.10509*, 2019.

[62] Sanh, V., Debut, L., Chaumond, J., and Wolf, T., "DistilBERT, a Distilled Version of BERT: Smaller, Faster, Cheaper and Lighter," *NeurIPS EMC^{2}Workshop*, 2019.

[63] Parisi, G. I., Kemker, R., Part, J. L., Kanan, C., and Wermter, S., "Continual Lifelong Learning with Neural Networks: A Review," *Neural Networks*, Vol. 113, 2019, pp. 54–71.

[64] Jin, Y., Liu, B., Wang, Z., Chen, Y., Mao, J., Liu, Y., Zhang, M., and Ma, S., "Incorporating External Knowledge into Machine Reading for Generative Question Answering," *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*, 2019, pp. 2243–2253.