Graph Neural Networks for Natural Language Processing: A Systematic Literature Review

Nicola Atti
nicola.atti@studio.unibo.it

February 6, 2023

Contents

1 Introduction

Deep learning has recently begun to have great success in numerous fields of artificial intelligence, particularly in that of natural language processing (NLP). Traditionally, text inputs are represented as token sequences, with models such as bag of words, a representation of text that describes the occurrence of words within a document that keeps track of word counts and disregards the grammatical details, word order and location, word structure and its semantics; often along with some kind of scoring metric like TF-IDF, a numerical statistic intended to reflect how important a word is to a document in a collection or corpus by increasing proportionally to the number of times a word appears in a document, but is offset by the number of documents that contain the word. So, words that are common in every document, rank low even though they may appear many times, since they don't mean much to that document in particular.

However, there is a huge variety of NLP problems that can be expressed with a graph structure, indeed popular deep learning techniques such as Recurrent Neural Networks (RNN) and Convolutional Neural Networks (CNN) have begun to generate a lot of interest for the realization of new deep learning techniques based on graphs, namely Graph Convolutional Networks (GCN) and Graph Recurrent Neural Networks (GRNN). These new deep learning techniques have been used in a large variety of graph-related problems, including NLP problems as the sentence structural information in text sequence can be exploited to augment original sequence data by incorporating the task-specific knowledge. Similarly, the semantic information in sequence data can be leveraged to enhance original sequence data as well.

The objective of this systematic literature review (SLR) is to investigate which new techniques have been developed in the context of knowledge injection that is, the act of combining the purely data-driven learning of neural networks with the infusion of knowledge from external sources, focusing on NLP applications and examining the pros and cons compared to the approaches considered traditional.

The work will be structured as follows: section 2 describes and documents the research method used to obtain the sources, section 3 reports the results obtained by discussing them and answering the research questions. Finally, section 4 will present the conclusions.

2 Method

This Systematic Literature Review investigates knowledge injection relating NLP tasks and is performed in three phases: planning, conducting and writing the review.

In this section will be presented the steps to follow in order to perform the review and reduce the researcher bias. These include:

1. Identifying the research questions.
2. Define the search strategy.
3. Define the study selection criteria.

4. Assess the quality of the selected studies.

2.1 Research Questions

The research questions that have given rise to the need for this study are:

- What are the NLP tasks where graph based deep learning techniques are used?
- How can the different techniques be classified?
- What advantages and disadvantages the different techniques have in relation to traditional methods?
- How are graph neural networks used for knowledge injection? And what kind of knowledge?

2.2 Source

The search process was conducted using some of the most relevant digital libraries.

- Google Scholar.
- Semantic Scholar.
- Scopus.

Other sources such as *IEEE Xplore* or *Dblp* were consulted, but offered too few results and mostly duplicates, and therefore won't be reported in the following result tables.

In order to perform our research, multiple keywords have been identified by doing a preliminary study of some related articles. The resulting keywords are:

- **NLP**, natural language processing
- **GNN**, graph neural network
- **GCN**, graph convolutional network
- Graph embedding
- Knowledge base completion

For obvious reasons NLP and GNN are two perfect candidates for keywords as they represent the main topics for our research and have been kept as constants in all our research queries. Initially, to tighten the scope of our research, studies were provided by the professor in order to get a better understanding and prior knowledge on the topic of knowledge injection. Thanks to this we were able to identify additional keywords, representing techniques or tasks inherent to knowledge injection in NLP.

2.3 Search Strategy

During our research no constraints were imposed regarding the publication date of the articles, as the topic is fairly recent and restricting an already small pool of sources may have produced too few results. Therefore any article whose publication date is between 1960 and 2022 was considered valid for the research. Indeed, as can be seen later, the works identified were all carried out in the last five years. The documents that have been taken into consideration are those written in English.

The sources were interrogated by composing five different queries. Each contains the keywords NLP and GNN, in order to keep the results from drifting away from the given topic, plus additional keywords in order to focus on a specific sub-topic. The queries that have been used are:

- Nlp *AND* Gnn
- Nlp *AND* Gnn *AND* Gcn
- Nlp *AND* Gnn *AND* Graph *AND* Embedding
- Nlp *AND* Gnn *AND* Knowledge *AND* Aware *AND* Model
- Nlp *AND* Gnn *AND* Knowledge *AND* Base *AND* Completion

After some initial research, it was noticed that after the first 100 results the correlation with the given topic decreased drastically, so the number of results for each library has been limited to 100.

	Semantic Scholar	Google Scholar	Scopus
Nlp Gnn	5250	6300	28
Nlp Gnn Gcn	343	3350	1
Nlp Gnn Graph Embedding	1030	5760	7
Nlp Gnn Knowledge Aware Model	172	3760	1
Nlp Gnn Knowledge Base Completion	87	3540	13

2.4 Study Selection

In order to identify the primary works to analyze, we applied a filtering process on the search results, carried out by defining inclusion and exclusion criteria.

This process has reduced the list of papers to the most relevant, that will be further analyzed. The criteria that have been used are:

- Inclusion criteria
  - Full paper
  - Articles published in peer-reviewed journals or conference proceedings
  - Sources describing the research topic

  - Sources answering the research questions

- Exclusion criteria
  - Documents with limited access (or not accessible with university credentials)
  - Papers that do not discuss the topic of this SLR inside the abstract
  - Papers that do not answer any of the research questions
  - Duplicate reports
  - Whole books

In addition to the documents obtained from the selection process, the papers provided by the professors and those used for the identification of the keywords, which did not appear during the research using the search engines, were also added.

At the end of the source selection process, the number of documents left was 43.

2.5 Quality Assestment

The remaining papers have been examined in depth, and this caused the removal of some documents. At the end of the evaluation process the amount of primary studies amounted to 39.

3 Results

In this section will be summarized the results of the analysis of the final list of papers. The section is organized in four parts, each answering one of the research questions.

3.1 Question 1. What are the NLP tasks where graph based deep learning techniques are used?

As already mentioned in the introduction, deep learning techniques based on graphs have begun to take hold in numerous fields of NLP, in particular some works such as [33] and [14] discuss a large variety of applications of GNNs for NLP tasks, demonstrating how these techniques are becoming more and more adopted in this field.

The purpose of this section is to list, in no particular order, the NLP problems faced by the works found during our research, describing how graph-based techniques have been employed for their resolution.

3.1.1 Sentiment Analysis

Sentiment analysis (or opinion mining) is a technique used to determine whether text data (eg. reviews, survey responses, social media posts) have a positive, negative or neutral sentiment polarity. As an example a sentence such as "I really liked the place" will almost certainly be positive, while the sentence "Really bad experience, I don't recommend it to anyone" will most likely be labeled negative.

As stated by [13], traditional methods of performing this task have shown particular flaws:

- Dictionary-based, where a sentiment vocabulary is built manually, has it's results strictly tied to the quality of the dictionary.
- Supervised machine learning, where the text is manually labeled and a statistical model is built through training, depends on the quality of feature selection.
- Deep learning, although shows good feature extraction ability, often ignore the overall structure of sentences.

GNNs can improve on the standard deep learning techniques by capturing structural characteristics. Applications in this field can be found in studies as [22], in which a text graph, based on the co-occurrence relationship between words, is built and then enhanced by integrating syntactic information. Then a Gated Graph Neural Network is used, to learn the embedding of the word nodes, in conjunction with an attention mechanism to retain the most relevant text features.

Algorithm 1 Assigning affective polarity to hashtags

Require: Hashtag h_0 coded as a case-sensitive string
Ensure: Sentiment polarity of h_0 is computed
1: set counter c_0 to zero
2: **for all** words w of h_0 **do**
3: **if** w is positive according to SentimentAnalysis **then**
4: add one to c_0
5: **else if** w is negative **then**
6: subtract one from c_0
7: **end if**
8: **end for**
9: **return** $\operatorname{sgn}(c_0)$

(a) A subfigure

Algorithm 2 The proposed GNN

Require: State vectors as in equation (1)
Ensure: Robust affective polarity labels $\{l_k\}$
1: obtain intial labels $\left\{l_k^{[0]}\right\}$ from algorithm 1
2: **repeat**
3: **for all** vertices v_k **do**
4: update local label l_k as in equation (3)
5: **end for**
6: **until** the relative steady state difference drops under η_1
7: **return** labels $\{l_k\}$

(b) A subfigure

Figure 1: The proposed GNN works iteratively. The relative steady state difference will track convergence. It is the number of labels which have changed more than a threshold $\eta 0$ between two iterations to the number of labels. When it is under a threshold $\eta 1$, the iteration terminate

Other works like [7] propose an iterative GNN architecture that locally aggregates vertex state vectors containing structural, functional and affective attributes in order to better capture emotional polarity from Twitter posts. How they achieve this classification is shown in figures 1a and 1b

Works like [13] propose a different approach, trying to mimic humans' ability to understand language at different levels, proposing a Multi-level Graph Neural Network that builds a graph from each single input text, employing multiple levels to capture local, long-distance and global features as show in figure 2.

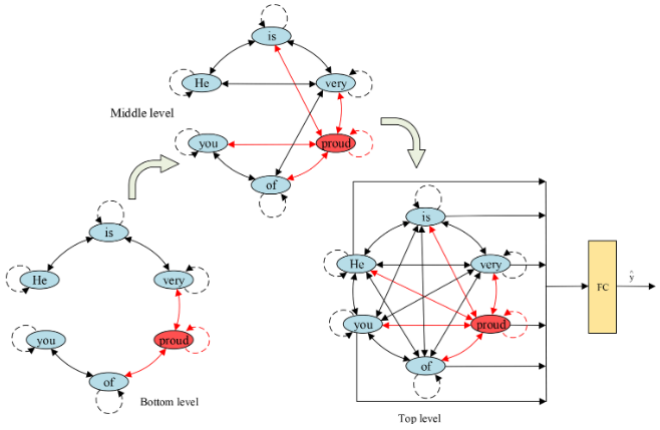

Figure 2: At the bottom level, features are collected from adjacent nodes, the node features are updated,then the node representations are converted into a new dimension d1. In the middle level, a larger node connection window is used. Lastly, to grasp the features of the full text, all the nodes are connected to each other at the top level

3.1.2 Multi-Document Summarization

Document summarization aims to produce coherent summaries starting from the information contained within documents.

According to [37] neural approaches have already been used in linked tasks such as sentence compression and single-document summarization, but have still to deal with multi-document summarization.

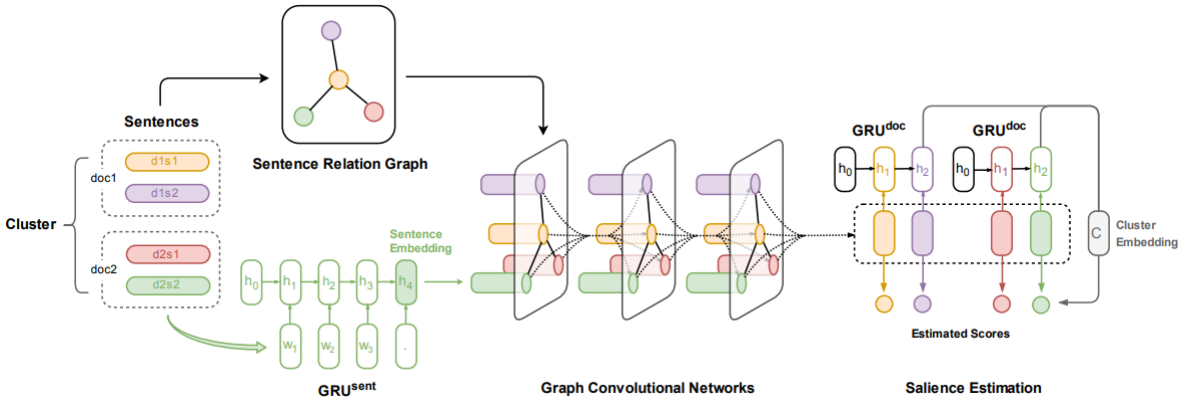

Figure 3: Illustration of our architecture for sentence salience estimation.

A solution is proposed by the authors which, given a document cluster, extracts a summary in two steps called: *sentence salience estimation* and *sentence selection*. During the phase of salience estimation, for each sentence an RNN with Gated Recurrent Units

is applied to extract the sentence embeddings, then a GCN is applied to the sentence relation graph to produce the final sentence embeddings that reflect the graph representation. Then a second level GRU produces the entire cluster embeddings. Finally after estimating the salience of each sentence, from the sentence embeddings and the cluster embeddings, sentences are selected in a greedy way until a maximum length is reached, to obtain the final summarization. In order to perform sentence selection, given the salience score estimation, they apply a simple greedy procedure to select sentences until a lenght limit is reached. Sentences with higher salience scores have higher priorities.

3.1.3 Fake News Detection

Multiple works have been found that tackle the problem of detecting fake news by making use of GNNs. In [10] and [24] GNNs have been adopted to solve the 2020 MediaEval task, aimed to the classification of different types of misinformation text regarding the COVID-19 pandemic.

Also in [23] the authors decide to use GNNs as a classifier to solve the task of Structure-Based Fake News Detection, specifically a Graph Isomorphism Network (GIN). After constructing the graph with the method proposed in [40], they decide to employ two GIN convolutional layers followed by a pooling layer (using sum) and fully connected layers. The embeddings are extracted from the last hidden layer and concatenated with the graph embeddings obtained just after pooling layer. They are sent to a MLP whose output layer will return the predictions for classification task.

And finally in [20] the DEAP-FAKED framework is proposed, combining NLP and GNN techniques. Here GNNs are used to encode the named entities, extracted from news text, into a knowledge graph (KG) in order to leverage additional information, in addition to the title, to further improve detection performance.

3.1.4 Machine Reading Comprehension

GNNs have also been successfully applied to tasks that require reasoning, such as machine reading comprehension. This task consists in training a model in understanding questions and formulating answers based on unstructured text.

In [28] a new type of GNN is proposed, called Graph Sequential Network (GSN), that instead of summarizing sequences into vectors and then using them for graph node initialization, directly takes sequence features as graph node representation. Furthermore it's co-attention based message passing mechanism can learn neighbour-aware representations of the current node. This strengthens the model ability to reason over sequences, and also makes sequential labeling tasks possibile, which is still impossible for other GNNs variants according to the authors.

This solution has also found application within the task of fact verification.

3.1.5 Text Representation

As stated by the authors of [39] methods for encoding sentences such as bi-directional LSTMs (BiLSTM) have been a dominant technique, but with several limitations such as

being prone to becoming a computational bottleneck and lower performances in encoding longer sentences.

In [39] a new model is proposed, sentence-state LSTM (S-LSTM), which is able to return an effective sentence encoding after 3 or 6 recurrent steps, as opposed to BiLSTM which scales with sentence size. S-LSTM uses a recurrent state transition process to model information exchange between sub states, which enriches state representations incrementally. The difference between S-LSTM and BiLSTM can be understood with respect to their recurrent states. While BiLSTM uses only one state in each direction to represent the subsequence from the beginning to a certain word, S-LSTM uses a structural state to represent the full sentence.

Other works like [21] decide to improve on the concept of GNN, proposing a Graph Transformer Networks based Text representation (GTNT) model, shown in figure 4, in order to solve the problems of high memory consumption and containing the test documents inside the training graph, also while taking into account word ordering. GTNT uses Graph Attention Networks (GAT) as the attention mechanism inside it's graph transformer layer. Since GAT considers all nodes to be equally influential with each other, but the authors believe that more important nodes should exert more influence on their neighbors, they added a value of *relative importance* when calculating the attention coefficient.

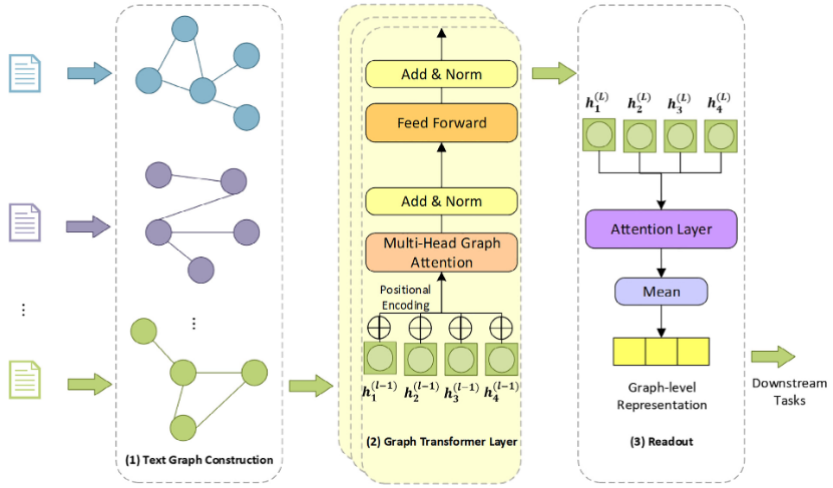

Figure 4: The architecture of GTNT.

3.1.6 Document Dating

Graph Convolutional Networks (GCN) have been used for solving the document timestamping problem, namely the problem of predicting the date of a document based on its contents. The need to incorporate the structure of the documents is addressed in [29], where they propose NeuralDater which uses a Bi-direction LSTM to capture the contextual embedding and two GCN layers to capture the syntactic and temporal embeddings,

and finally predict the dates and classify the documents.

3.1.7 Semantic Role Labeling

Semantic Role Labeling is the task of identifying the predicate-argument structure of a sentence.According to [19] the semantic representations are closely related to syntactic ones, and so propose a GCN based approach that also relies on syntax for its predictions, using a syntax-based GCN, on top of a BiLSTM encoder, that re-encodes the representation based on the automatically predicted syntactic structure of the sentence.

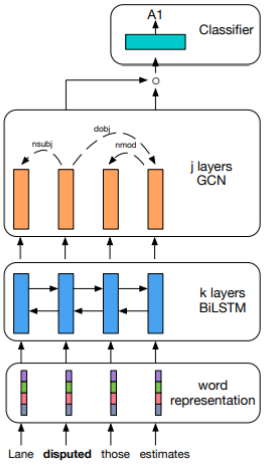

Figure 5: LSTM GCN encoder for semantic role labeling.

Furthermore, also in [29] two solutions, one called SynGCN which aims to encode syntactic knowledge in embeddings, and another called SemGCN that incorporates semantic knowledge into pre-trained word embeddings, are proposed to tackle the problem. The study shows how the combination of the two models offers the best performances.

3.1.8 Event Detection

In [9] a method based on GNN is proposed, called EDGNN, for event detection on social networks. This implementation aims to solve the problems of the classic GNN approaches to the task, that is the large memory consumption for the construction of the graph and the poor performance on short texts which make up the majority of the documents analyzed on social media. The process is divided in three steps, at first a Conditional Random Field regularized Topic Model (CRFTM) is employed to extract the topic information from text, the second step is to build a text-level graph. Unlike the previous graph-based methods of building a graph on the whole corpus, EDGNN is able to greatly reduce the nodes and edges of the graph.

After constructing the text-level graph, a GCN is used to train the proposed model. Firstly, the message passing process of the approach receives information from each

neighboring node. Secondly, the representation of each node is updated according to the original representation and received information. The message passing process makes the update of a node representation to be affected by its neighbors, which enables the representation to gather information from the context. As a result, the proposed method can capture their accurate meanings from their adjacent words and topics, also introducing global information during the training process.

As the last step the word and topic representations generated by the graph and BERT (which is used to enhance the performance) are used as input to a BiGRU model for event detection.

3.1.9 Text Classification

Many works have been found that deal with the problem of text classification, the problem of processing text labels into categories.

In [17] and [25] we find surveys regarding the adoption of graph neural networks and graph convolutional networks in the field of text classification. Particularly in the latter the authors give an overview on how GCNs can convert text into a graph and how they can be employed in the field of text classification and sentiment analysis, while in the former they talk in depth about specific solutions that aim to outperform traditional approaches, such as:

- Text GCN, where a two layer-GCN is employed on a pre-composed graph.
- Text Level GNN, that generates graphs for each input text. This technique drops the dependency between individual text and the whole corpus, allowing for online testing and preserving global information.
- MAGNET, which uses a Graph Attention Network (GAT) to determine the importance of labels in a correlation graph, a BERT model to obtain the embeddings of the words and a LSTM to get the feature vector.
- STGCN, a GCN specialized in classifying short text, with an architecture composed of a two layer GCN followed by a softmax layer.

Other studies propose their solution to the problem by developing new techniques and architectures.

In [3] the authors describe an approach to perform zero-shot text classification called S-BERT-KG. Using an external knowledge graph (i.e ConceptNet) they build a knowledge graph embedding space. By embedding the graph structure into the KG they claim to create a semantic space more effective in terms of prediction, classification and recommendation performance.

The authors of [16] proposes another BERT augmentation, called VGCN-BERT, where a graph convolutional network is used on a vocabulary graph, and then the graph and word embedding together are fed to a self-attention encoder in BERT. This allows the classifier to use both local and global information.

A graph transformer network based architecture can be found in [21], which converts each input document into a text-level graph that can be used for inductive learning and then learns representation of each text-level graph.

In [31] a model to address the problems of standard RNNs and GNNs is proposed and applied to the context of text classification. The structure of the model is displayed in figure 6 where firstly they apply a BiLSTM to capture the full contextual information, then by using the message passing strategy of GGNNs to aggregate the most important features from their neighbors and fuse them into its own ones. By using this method, each word's representation can better aggregate important information and ignore less important information. Lastly a pooling layer is applied that automatically judges which feature plays a critical role in text classification, as opposed to traditional RNN-based models that just use the last hidden state as input for the classifier, which has a high probability of ignoring some relevant information.

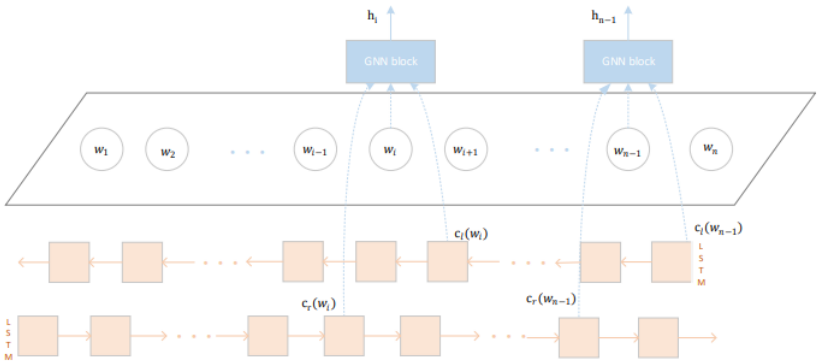

Figure 6: The structure of the recurrent graph neural network. The orange block is the recurrent structure that calculates right and left contextual information, and blue block denotes graph neural network update block.

According to [34] many of the existing graph-based models only consider either local word relations or global co-occurrence relations. In their work the authors propose a two-window graph construction method (TW-TGNN) that uses a local sliding window, for determining the connection relationship between nodes, and a dynamic global window that generates the corresponding dynamic shared matrix, which can obtain the edge weight. The graph structure is then fed to a GGNN message passing (that adopts Gated Recurring Units) model to update node representation by aggregating the neighbor node information. Lastly the nodes representations will be aggregated to a graph-level representation that will be used to predict the label of the text.

A new transformer-based solution, Text Graph Transformer (TG-Transformer), presented in [38] aims to resolve the scalability problems of standard GNN approaches on a large corpus. As shown in figure 7, the model starts with a graph construction module, containing word and document nodes (representing all documents in the corpus and all the words in the corpus vocabulary respectively). The graph also contains two types of edges: word-document edges built based on word occurrence within documents and

word-word edges based on local word co-occurrence within sliding windows in the corpus.

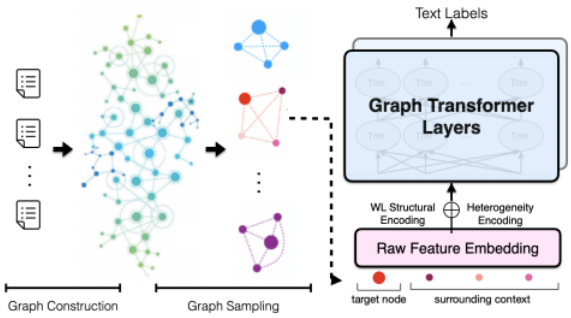

Figure 7: Overall Structure of TG-Transformer.

The second step, called graph sampling, consists in the training of TG-Transformer on sampled subgraph mini-batch. Based on the sampled subgraph mini-batch, TG-Transformer will update the text graph nodes' representations iteratively for classification.

Lastly in [40] a text classification method for inductive word representation using GNNs is proposed, named TextING. Here a GNN is trained to capture word-word relations by only using training documents. Then the information of word nodes is propagated using GGNNs, which is then aggregated to the word embedding.

3.1.10 Natural Language Inference

The goal of Natural Language Inference is the task of determining whether a "hypothesis" is true, false, or undetermined given a "premise".

The possibility of adding pre-existing knowledge to this problem has only recently been taken into consideration. In [30] we find one possible approach, where they propose a framework called ConSeqNet, which is able to use both textual as well as structured information from multiple knowledge bases to determine textual entailment. The framework is divided into two parts:

- Text Based Model: the premise and hypothesis are encoded using recurrent neural networks. Next, an attention layer is implemented on top of the encoders. The final layer is then used for classification. The authors opt to use match-LSTM as the entailment model, as its been proved useful for multiple NLP tasks.

- Graph Based Model: captures structured information by taking in input a premise and an hypothesis graph. Premise and hypothesis graphs are generated by mapping text phrases to concepts in the external knowledge graph. The embeddings of premise and hypothesis concepts from the graph are the input to the neural network.

The final results of matching the text model and the graph model are concatenated, and passed on to a feed forward network that classifies between *entailment* and *neutral*,

which are the two classes in the SciTail dataset (which is the dataset used by the authors during the experimentation phase).

3.1.11 Neural Machine Translation

Recent applications have been made in the field of Neural Machine Translation, the task of translating text from one language to another preserving the meaning of the input text without human supervision, mostly by injecting syntactic structure. Works like [18] take a different approach, by considering semantic structure instead, applying GCNs to the semantic dependency graphs and experimenting on the English-German language pair.

3.1.12 Text to Speech

Applications have been found even in the context of Text to Speech, in particular to improve the problem of prosody when reproducing text, as shown in [26]. They propose GraphTTS, which uses GCN or GGNN as the graph encoder which receives all nodes states as source states and delivers messages according to the adjacency matrix to aggregate messages to target nodes. The output of the propagation model is transformed by the output model to a graphical hidden state.

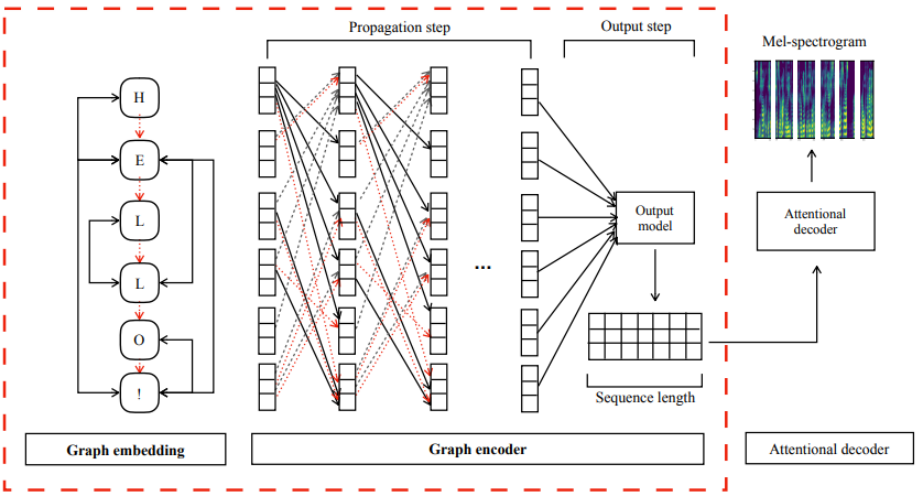

Figure 8: GraphTTS model overview.

3.1.13 Question Answering

Some works have been found that propose solutions in the context of question answering (QA).In [5] an approach for multi-hop QA is proposed which, instead of treating all documents as a single one (composed concatenating each one), represents the document collection as a graph. Each node corresponds to named entities in a document,

and the edges encode relations between them. Relational-GCNs are used to propagate information through the entity graph, updating a node from it's direct neighbours at each step.Each step of the algorithm (also referred to as a hop) updates all node representations in parallel.In particular, a node is updated as a function of messages from its direct neighbours, and a message is possibly specific to a certain relation.

Also the authors of [8] propose Multi-hop Graph Relation Network (MHGRN) that aims to merge GNNs and path-based models, with the former granting better scalability and the latter granting path-level reasoning and interpretability, in order to surpass traditional GNN approaches.

The authors of [36] aim to realize a new model to perform reasoning on both structured and unstructured knowledge for question answering systems. They propose QA-GNN, a model fusing language models (LM) and knowledge graphs. The proposed implementation follows two key insights:

- Relevance scoring: where each entity of the KG subgraph is scored using the QA context and a pre trained LM, presenting a general framework to weight information on the KG.
- Joint reasoning: where a joint graph is designed with an additional node representing the QA context connected with the topic entities of the KG subgraph.

This final graph, which they call working graph unifies the two types of knowledge, and it's used by an attention-based GNN module to perform reasoning.

Finally in [27] a framework, called Pullnet, is proposed to extract information from both the text and KB, and combine them into a single data structure to allow systems to reason and find better answers to questions. Pullnet uses CNN to reason over a graph, built by a set of retrieval operations carried out on a pre-existing graph. By learning when and where to apply this operations it also learns how to build a subgraph that starts with only the question text and its entities, and ends by containing additional information from the KBs and the corpus.

3.1.14 Sentence Insertion

The problem with sentence insertion is deciding where is the best place to put a sentence within a text. Although not a hot topic in NLP, [32] propose a solution that aims to emulate the way that humans tackle the TOEFL (Test of English as a Foreign Language) and using it as a benchmark. InsertGNN (shown in figure 9) is proposed, where at first it splits the paragraph into five parts, according to positions of four different slots (A, B, C, D), and feed them to a sentence encoder, obtaining representation vectors for each sentence in the paragraph (Sc1, ..., Sc5) and Sq for the taken-out sentence. These vectors correspond to node features, where Sq corresponds to features of node (A, B, C, D).

Then it applies a L-layer Global Graph Neural Network (GGN) with attention is applied to extract the global information. As GGN may ignore local details, a Local

Graph Neural Network is used to focus on the local sentence interactions. Four subgraphs are created with only the slot and its two surrounding sentences, whose node features are from the output of the previous GGN.

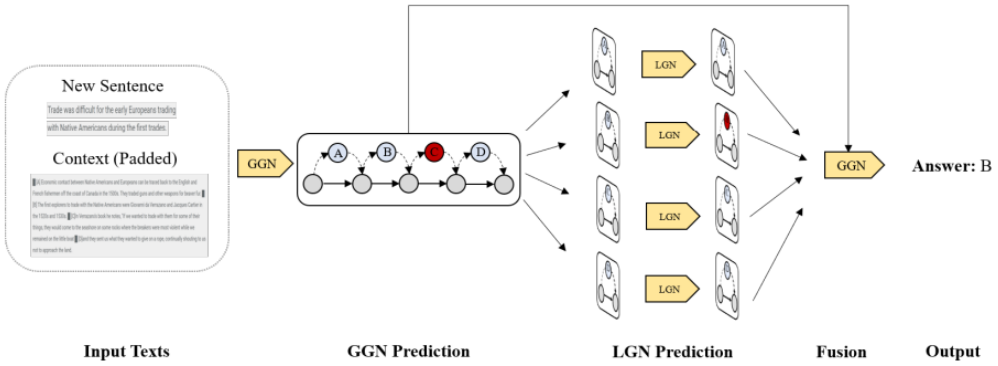

Figure 9: InsertGNN architecture.

At the final stage both views are combined and fed to a Multi Layer Perceptron (MLP) to obtain the final prediction.

3.1.15 Text-Based Relational Reasoning

In this task a system needs to infer relations between entities relying on a passage of text. The authors of [6] propose an approach to *distill* structured knowledge learned by a GNN to improve the performance of an NLP model.

The authors start by implementing both a graph encoder using a variant of the graph isomorphism network (GIN), which generates a vector embedding based on the input family graph, as well as a text encoder which generates a vector embedding of the input text, experimenting with two top performing NLP models: a variation of an LSTM model with attention and an adapted version of the MAC architecture.

Then they utilize knowledge distillation for the structured knowledge transfer from GNNs to NLP models, pairing it with a mutual information (MI) based contrastive learning method to maximize the MI between graph representations, to mitigate the lack of regularization between the latent representations of the NLP and GNN modules.

3.1.16 Knowledge Graph Completion

GNNs have also been used in the field of knowledge graph completion, the problem of extending a knowledge graph by adding missing facts.

In [4] the authors propose a new GNN architecture, called Monotonic Graph Neural Network (MGNN), that processes a dataset, that has been encoded into a graph whose vertices are labeled with numeric feature vectors, decoding it into an output dataset from which will be extracted a set of rules that will complete the given KG. MGNN

are used because a normal GNN cannot mimic Datalog rule application, which is the objective of the work.

3.2 Question 2. How can the different techniques be classified?

This section will try to classify the techniques used by the works detected in the research phase, also going to briefly analyze some of the innovative ideas found.

Initially, papers such as [42] and [35] were analyzed, which provide an important initial classification of the main methods used so far in the context of knowledge injection. From these, an initial classification was created which influenced the research and evaluation process.

	Works
GNN	[7] [10] [13] [11] [8] [24] [38] [20] [36] [37]
GCN	[19] [9][18] [37] [26] [11] [29] [27] [5] [25] [16]
GGNN	[40] [34] [26]
GIN	[6] [23]
LSTM	[20] [29] [39] [30]
GAT	[36] [21] [15]
MPNN	[34]

Table 1: Classification of the works found according to the topics covered

The classification of the examined documents can be found in table 1. An important note is that since normally a GNN architecture works by combining different architectures and models between the propagation, sampling and pooling modules in conjunction with each other, or some articles apply different solutions during their experiments, each document can appear in multiple categories depending on the covered models.

By examining the final works we found that the most used architecture are GNNs and GCNs, or some minor variation specifically created to tackle a specific issue as in [37], [32], [16] and [13].

Some of the works also propose totally new solutions, not described in the examined surveys. [28] proposes GSNs, that also implement a two-step computation process composed by aggregation and combination. The former calculates structure-aware feature representations from neighbours of the target node, and the latter fuses the representations obtained during the aggregation phase with the current node's feature representation.

In [15] RAGAT defines new relation aware message passing functions, expanding on existing message functions, such as: SUB-GAT, MULT-GAT, CORR-GAT, CONCAT-GAT, CROSS-GAT and an algebraic perspective on CrossE.

The authors of [30] explore two possible neural network models for their solution:

- Gmatch-LSTM, where match-LSTM is applied with *Concepts Only* graphs, treating each node as a single token and re-arrange them into a sequence of concepts.

- GconAttn, designed to be used with generic graphs, determines and transforms an attention weighted representation of premise and hypothesis into a final representation using two-way attention and a max-pooling layer.

And finally the solution proposed in [4] specializes standard GNNs by:

- Taking the maximum of the features of a vertex neighbours, instead of the sum or average.
- Using positive weights in the matrices that define each layer.
- Having the activation and classification functions monotonically increasing.

3.3 Question 3. How are graph neural networks used for knowledge injection? And what kind of knowledge?

Graphs have been widely used in natural language processing because of their potential to represent text items and relations. The nodes and edges in the graph can represent various entities and links depending on the NLP specified application. Generally, nodes represent basic text units, e.g., words, word senses, entities, paragraphs, or documents. Edges, however, are usually used to describe relationships like syntactic structure, semantic relations, lexical similarity, or other knowledge-based links.

These relationships can be used as background knowledge in learning algorithms in order to inject knowledge, which would normally be ignored, but which a human being would normally know. This can be seen in many of the works identified during the research phase of this SLR.

In works such as [3] and [8] they use external knowledge bases to inject common sense knowledge and domain specific knowledge. Both works leverage ConceptNet, a knowledge graph that aims to represent the common sense knowledge involved in understanding language and to improve natural language applications by enabling applications to better understand the meaning behind the words.

Incorporating KGs brings the potential of interpretable and trustworthy predictions, as the knowledge is now explicitly stated. For example, figure 10 shows how the external knowledge is used by [8], where the relational path (CHILD $\rightarrow$ AtLocation $\rightarrow$ CLASSROOM $\rightarrow$ Synonym $\rightarrow$ SCHOOLROOM) naturally provides evidence for deduce the answer SCHOOLROOM.

The authors of [30] exploit external knowledge bases such as WordNet, ConceptNet and DBPedia to inject background knowledge. Their solution, ConSeqNet, takes in as input the premise and hypothesis text, and a graph based model whose input is specific knowledge derived from the knowledge base using the given premise and hypothesis. The input to the graph based model is a premise graph and a hypothesis graph respectively. Premise and hypothesis graphs are generated by mapping text phrases to concepts in the external knowledge graph. The embeddings of premise and hypothesis concepts from the graph are the input to a neural network. An attention weighted representation of both premise and hypothesis is determined, and is transformed to a final fixed size

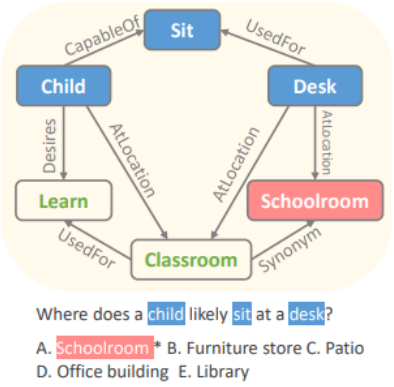

Figure 10: Illustration of knowledge-aware QA. Blue nodes correspond to entities mentioned in the question, pink nodes correspond to those in the answer.

representation using pooling. This representation of premise and hypothesis is then used for classification.

The solution proposed in [36] also combines language models and knowledge graphs in order to solve the question answering problem. This is shown in figure 11 where given a QA context methods need to identify informative knowledge from a large KG and capture the nuance of the QA context and the structure of the KGs to perform joint reasoning over these two sources of information. To design a joint reasoning space for the two sources of knowledge, they explicitly connect them in a common graph structure. To perform reasoning on the working graph, the GNN module builds on the graph attention framework (GAT), which induces node representations via iterative message passing between neighbors on the graph. Crucially, as the GNN message passing operates on the working graph, it will jointly leverage and update the representation of the QA context and KG.

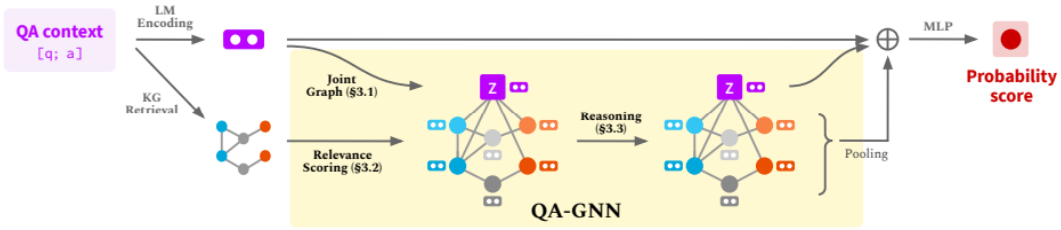

Figure 11: Given a QA context , is connected with the retrieved KG to form a joint graph, the relevance of each KG node is computed and conditioned on the context, and then reasoning on the working graph is performed.

Other works aim to first create a knowledge graph (starting from the already available text data) and then make use of mechanisms typical of graph neural networks to propagate the distilled information through the used models and NLP techniques. In [20]

the proposed model identifies and extracts named entities from the news text and then maps them to a KG. Next, it employs a GNN-based technique (which they call Entity Encoder) to encode the entities in the KG in order to consider not only the news content but also the association of multiple entities present in the news. This approach was also found in [16] where a vocabulary graph is constructed using word co-occurrences with documents, thus forming the global language information, while the local language information is being captured by BERT. The interaction between this two forms of information is achieved by first selecting the relevant part of the global vocabulary graph according to the input sentence and transforming it into an embedding representation, as show in figure 12.

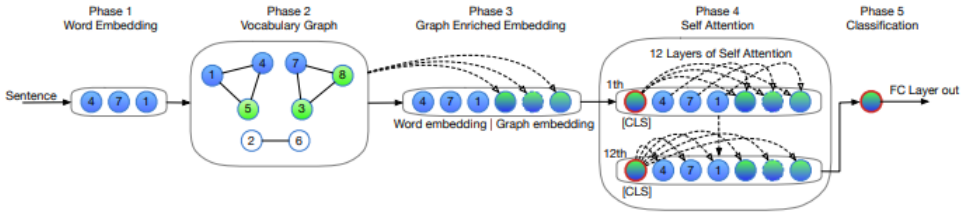

Figure 12: Illustration of how VGCN-BERT combines global and local language information to classify text.

The embeddings of the input sentence are combined with the vocabulary graph to produce a graph embedding concatenated to the input sentence. Then several layers of self-attention are applied to the concatenated representation, allowing interactions between word embeddings and graph embedding. The final embedding at the last layer is fed in a fully connected layer for classification.

3.4 Question 4. What advantages and disadvantages the different techniques have in relation to traditional methods?

In this section we will discuss the improvements that these techniques offer compared to the traditional approaches, and going where possible to list any limitations or problems that have emerged.

Almost all the works show some form of improvement, even significant, particularly the more recent or ad-hoc solutions, that better exploit the techniques, are able to outperform most of the state of the art NLP methods and also some previous uses of the model. In particular the authors of [13] report how much GNN based methods can improve the performance of text sentiment analysis, testing their model against previously enstablished methods. The results are shown in figure 13.

The experiments carried out by the authors of [20] show a 24% increase in terms of F1-score and 21% in terms of average accuracy on the KFN-UB dataset and 3% improvements on the CoAID-UB dataset. In addition, a minimal change in standard deviation is reported, which demonstrates the stability of the model.

Method	Dim.	SST-binary	SenTube-A	SenTube-T
BOW	–	80.7	60.6	66
AVE	50	74.1	62	61.7
	100	76.7	61.5	61.8
	200	78.2	60.6	62.8
	300	80.3	61.5	64.3
LSTM	50	80.5 ± 0.4	58.9 ± 0.8	63.4 ± 3.1
	100	79.5 ± 0.6	58.9 ± 1.1	63.1 ± 0.4
	200	80.9 ± 0.6	58.6 ± 0.6	65.2 ± 1.6
	300	81.7 ± 0.6	57.4 ± 1.3	63.6 ± 0.7
BiLSTM	50	82.9 ± 0.7	59.5 ± 1.1	65.6 ± 1.2
	100	79.8 ± 1.0	58.6 ± 0.8	66.4 ± 1.4
	200	80.1 ± 0.6	58.9 ± 0.3	63.3 ± 1.0
	300	82.6 ± 0.7	59.3 ± 1.0	66.2 ± 1.5
CNN	50	81.7 ± 0.3	55.2 ± 0.7	57.4 ± 3.1
	100	81.6 ± 0.5	56.0 ± 2.2	61.5 ± 1.1
	200	80.7 ± 0.4	56.3 ± 1.8	64.1 ± 1.1
	300	81.3 ± 1.1	57.3 ± 0.5	62.1 ± 1.0
Huang et al.	50	82.5 ± 0.1	60.0 ± 0.3	67.3 ± 0.1
	100	82.6 ± 0.1	60.2 ± 0.2	67.3 ± 0.1
	200	82.5 ± 0.3	59.7 ± 0.1	67.1 ± 0.1
	300	82.6 ± 0.2	59.7 ± 0.1	67.4 ± 0.2
MLGNN	50	81.3 ± 0.5	61.5 ± 2.9	67.9 ± 0.5
	100	82.4 ± 0.5	61.2 ± 3.5	68.1 ± 0.5
	200	82.4 ± 1.1	61.5 ± 1.5	67.7 ± 0.6
	300	83.1 ± 0.4	62.2 ± 0.5	68.0 ± 0.4

Figure 13: Figure showing the improvements of MLGNN over existing methods for sentimenti analysis over various datasets. MLGNN outperformes all the other methods. Furthermore, MLGNN has the highest accuracy with 300-dimensional word embedding on SST-binary and SenTube-A, and with 100-dimensional embedding on SenTube-T.

In [9] CRFTM is compared with other topic models for their ability to detect hidden topics. In particular, the model is compared with LDA (Latend Dirichlet Allocation) one of the most popular algorithms for topic extraction. Figure 13 shows the differences in scores between them. Also in [5] the model Entity-GCN is put to the test with previous works as part of the Question Answering and has proven to be the best performing model, showing improvements of about 2%.

Significant improvements have also been achieved by [31] where both RCNN and RGNN show better results than traditional models such as LTSM on the task of document classification, with RGNN obtaining the best results overall. The results for each of the used datasets can be found in figure 14.

Other results can be found in [16] where the experiments show how much VGCN-BERT manages to surpass other models for text classification. The results can be found if figure 15 where we can see that VGCN-BERT outperforms all the baseline models (outside specific cases). In particular it outperforms both VGCN and BERT on their own, showing the advantage of combining them, indeed both Vanilla-VGCN-BERT and VGCN-BERT show better results than the baseline models and this can be attributed to the fact that both models combine local and global information. However, it is noted that the generalized version generally survives better, thanks to a different interaction between global and local information.

While the topic examined by this SLR is relatively new, many of the papers examined aim at improving graph-based techniques previously developed. The authors of [15]

Model	20NG	R8	R52	MR
CNN-rand	76.93	94.02	86.95	74.98
CNN-non-static	82.15	95.71	87.59	77.75
LSTM	65.71	93.68	85.54	75.06
LSTM (pretrain)	75.43	96.09	90.48	77.33
BiLSTM	73.18	96.31	90.54	77.68
RCNN	83.65	96.20	91.12	78.46
Text GCN	86.34	97.07	93.56	76.74
SLSTM	87.42	97.85	94.34	82.45
RGNN	**87.78**	**98.82**	**95.28**	**83.87**

Figure 14: The figure shows the test accuracy on the task of document classification for RGNN.

Model	SST-2	MR	CoLA	ArangoHate	FountaHate
MLP	80.78	75.55	61.39 (53.20)	84.71 (84.42)	79.22 (65.33)
Text-GCN	80.45	75.67	56.18 (52.30)	84.77 (84.43)	78.74 (64.54)
Bi-LSTM	81.32	76.39	62.88 (55.25)	84.92 (84.58)	79.04 (65.13)
VGCN	81.64	76.42	63.59 (54.82)	85.97 (85.69)	79.00 (64.04)
BERT	91.49	86.24	81.22 (77.02)	87.99 (87.75)	80.59 (66.61)
Vanilla-VGCN-BERT	91.38	**86.49**	80.70 (76.30)	88.01 (87.79)	81.11 (67.86)
VGCN-BERT	**91.93**	86.35	**83.68 (80.46)**	**88.43 (88.22)**	**81.26 (68.45)**

Figure 15: The figure shows the weighted average F1-Score and (Macro F1-score) on the test set for VGCN-BERT and the other baseline methods.

compare their solution with existing knowledge embedding methods, their results are reported in figure 17.

Although not quantifiable, it is noted that the use of GNN allows to capture hidden elements in the text, which would not have emerged through the use of traditional methods. Especially in works like [7] and [24] we note how GNNs manage to capture hidden elements of knowledge such as sarcasm or humor, or as in [29] we find that they are able to exploit the temporal elements within the text, that normally are ignored by standard NLP methods.

However, this type of architecture is not without its problems. Indeed, many of the works show that these introduce substantial problems of computational capacity, especially when dealing with large amounts of data, due to the recurring nature of the models.

Many of the works such [9] [40] manage to circumvent this problem by focusing on short texts, mainly obtained by analyzing social network texts or single sentences, others like [38] [34] [21] propose solutions with specific mechanisms to reduce it's impact on the solution's performances.

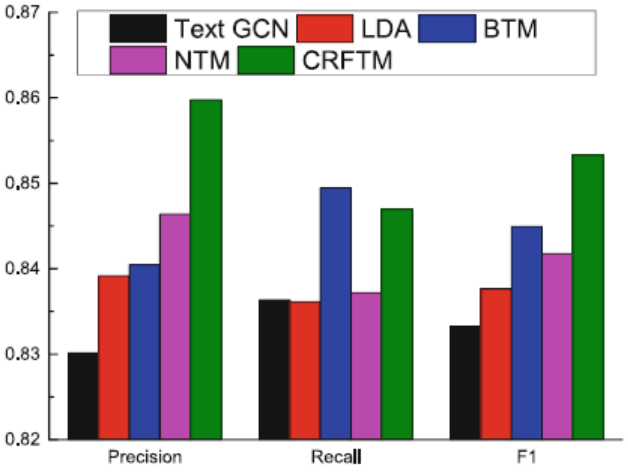

Figure 16: The figure shows the differences in terms of score for extracting hidden topics in tweets.

	FB15k-237					WN18RR				
	MRR	**MR**	**Hit@1**	**Hit@3**	**Hit@10**	**MRR**	**MR**	**Hit@1**	**Hit@3**	**Hit@10**
TransE [10]	0.294	357	-	-	0.465	0.226	3384	-	-	0.501
DistMult [15]	0.241	254	0.155	0.263	0.419	0.43	5110	0.39	0.44	0.49
ComplEx [17]	0.247	339	0.158	0.275	0.428	0.44	5261	0.41	0.46	0.51
ConvE [20]	0.325	244	0.237	0.356	0.501	0.43	4187	0.40	0.44	0.52
RotatE [42]	0.338	177	0.241	0.375	0.533	0.476	3340	0.428	0.492	**0.571**
ConvR [31]	0.35	-	0.261	0.385	0.528	0.475	-	0.443	0.489	0.537
R-GCN [21]	0.248	-	-	-	0.417	-	-	-	0.137	-
VR-GCN [24]	0.248	-	0.159	0.272	0.432	-	-	-	-	-
SACN [22]	0.35	-	0.26	0.39	0.54	0.47	-	0.43	0.48	0.54
CrossE [27]	0.299	-	0.211	0.331	0.474	-	-	-	-	-
KBGAT [25]	0.157	270	-	-	0.331	0.412	**1921**	-	-	0.554
InteractE [29]	0.354	**172**	0.263	-	0.535	0.463	5202	0.43	-	0.528
TransE-GCN [37]	0.315	-	0.229	0.324	0.477	0.233	-	0.203	0.338	0.508
G2SKGE [39]	0.342	-	0.253	0.374	0.515	0.447	-	0.424	0.467	0.493
A2N [38]	0.317	-	0.232	0.348	0.486	0.45	-	0.42	0.46	0.51
COMPGCN [26]	0.355	197	0.264	0.39	0.535	0.479	3533	0.443	0.494	0.546
RAGAT	**0.365**	199	**0.273**	**0.401**	**0.547**	**0.489**	2390	**0.452**	**0.503**	0.562

Figure 17: The figure shows the performance of RAGAT against existing knowledge embedding models. Compared to other baselines, RAGAT outperforms all other methods 4 out of 5 metrics on FB15k-237 and 3 out of 5 metrics on WN18RR, which indicates the whole effectiveness of our model.

4 Conclusion

This paper reports an SLR dedicated to analyzing current graph-based knowledge injection techniques applied to NLP, examining innovative solutions and analyzing their pros and cons compared to traditional approaches. It seemed quite evident that many NLP problems can be tackled from a graph perspective, and GNNs are naturally applicable to and good at handling graph-structured data. However, constructing a high quality and

task-specific graph still requires a good amount of domain expertise and human effort.

Our research shows that the practice of injecting knowledge into NLP models and the use of GNN and their techniques have brought tangible benefits and mark a clear step forward in the ability of artificial intelligence systems to understand and emulate knowledge and emotions, that humans naturally imprint in text which have still been ignored are a hot area of research. Also, given by the fact that most of the works that have been found were carried out over the last five years, this shows that the application of graph based models in the field of NLP to inject knowledge it is still in its infancy, and that it will surely bear further fruit over time.

During our research numerous articles were initially found by searching through digital libraries (Google Scholar, Google Scholar, Scopus) by running the queries presented. After examining selected articles and carrying out a pilot search, 39 articles were analyzed in depth.

References

[1] Daniel Beck, Gholamreza Haffari, and Trevor Cohn. Graph-to-sequence learning using gated graph neural networks, 2018.

[2] Tarek R. Besold, Artur d'Avila Garcez, Sebastian Bader, Howard Bowman, Pedro Domingos, Pascal Hitzler, Kai-Uwe Kuehnberger, Luis C. Lamb, Daniel Lowd, Priscila Machado Vieira Lima, Leo de Penning, Gadi Pinkas, Hoifung Poon, and Gerson Zaverucha. Neural-symbolic learning and reasoning: A survey and interpretation, 2017.

[3] Qi Chen, Wei Wang, Kaizhu Huang, and Frans Coenen. Zero-shot text classification via knowledge graph embedding for social media data. *IEEE Internet of Things Journal*, 9(12):9205–9213, 2022.

[4] David Jaime Tena Cucala, Bernardo Cuenca Grau, Egor V. Kostylev, and Boris Motik. Explainable GNN-based models over knowledge graphs. In *International Conference on Learning Representations*, 2022.

[5] Nicola De Cao, Wilker Aziz, and Ivan Titov. Question answering by reasoning across documents with graph convolutional networks, 2018.

[6] Jin Dong, Marc-Antoine Rondeau, and William L. Hamilton. Distilling structured knowledge for text-based relational reasoning. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6782–6791, Online, November 2020. Association for Computational Linguistics.

[7] Georgios Drakopoulos, Ioanna Giannoukou, Phivos Mylonas, and Spyros Sioutas. A graph neural network for assessing the affective coherence of twitter graphs. In *2020 IEEE International Conference on Big Data (Big Data)*, pages 3618–3627, 2020.

[8] Yanlin Feng, Xinyue Chen, Bill Yuchen Lin, Peifeng Wang, Jun Yan, and Xiang Ren. Scalable multi-hop relational reasoning for knowledge-aware question answering, 2020.

[9] Wang Gao, Yuan Fang, Lin Li, and Xiaohui Tao. Event detection in social media via graph neural network. In *Web Information Systems Engineering – WISE 2021: 22nd International Conference on Web Information Systems Engineering, WISE 2021, Melbourne, VIC, Australia, October 26–29, 2021, Proceedings, Part I*, page 370–384, Berlin, Heidelberg, 2021. Springer-Verlag.

[10] Abdullah Hamid, Nasrullah Shiekh, Naina Said, Kashif Ahmad, Asma Gul, Laiq Hassan, and Ala Al-Fuqaha. Fake news detection in social media using graph neural networks and nlp techniques: A covid-19 use-case, 2020.

[11] Zhichao Huang, Xutao Li, Yunming Ye, Baoquan Zhang, Guangning Xu, and Wensheng Gan. Multi-view knowledge graph fusion via knowledge-aware attentional graph neural network. *Applied Intelligence*, pages 1–20, 06 2022.

[12] Luis C. Lamb, Artur Garcez, Marco Gori, Marcelo Prates, Pedro Avelar, and Moshe Vardi. Graph neural networks meet neural-symbolic computing: A survey and perspective, 2020.

[13] Wenxiong Liao, Bi Zeng, Jianqi Liu, Pengfei Wei, Xiaochun Cheng, and Weiwen Zhang. Multi-level graph neural network for text sentiment analysis. *Computers & Electrical Engineering*, 92:107096, 2021.

[14] Xiaochen Liu, Yang Su, and Bingjie Xu. The application of graph neural network in natural language processing and computer vision. In *2021 3rd International Conference on Machine Learning, Big Data and Business Intelligence (MLBDBI)*, pages 708–714, 2021.

[15] Xiyang Liu, Huobin Tan, Qinghong Chen, and Guangyan Lin. Ragat: Relation aware graph attention network for knowledge graph completion. *IEEE Access*, 9:20840–20849, 2021.

[16] Zhibin Lu, Pan Du, and Jian-Yun Nie. Vgcn-bert: Augmenting bert with graph embedding for text classification. In Joemon M. Jose, Emine Yilmaz, João Magalhães, Pablo Castells, Nicola Ferro, Mário J. Silva, and Flávio Martins, editors, *Advances in Information Retrieval*, pages 369–382, Cham, 2020. Springer International Publishing.

[17] Masoud Malekzadeh, Parisa Hajibabaee, Maryam Heidari, Samira Zad, Ozlem Uzuner, and James H Jones. Review of graph neural network in text classification. In *2021 IEEE 12th Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON)*, pages 0084–0091, 2021.

[18] Diego Marcheggiani, Jasmijn Bastings, and Ivan Titov. Exploiting semantics in neural machine translation with graph convolutional networks, 2018.

[19] Diego Marcheggiani and Ivan Titov. Encoding sentences with graph convolutional networks for semantic role labeling, 2017.

[20] Mohit Mayank, Shakshi Sharma, and Rajesh Sharma. Deap-faked: Knowledge graph based approach for fake news detection, 2021.

[21] Xin Mei, Xiaoyan Cai, Libin Yang, and Nanxin Wang. Graph transformer networks based text representation. *Neurocomput.*, 463(C):91–100, nov 2021.

[22] Liyue Niu, Qiusheng Zheng, and Long Zhang. Enhance gated graph neural network with syntactic for sentiment analysis. In *2021 IEEE International Conference on Advances in Electrical Engineering and Computer Applications (AEECA)*, pages 1055–1060, 2021.

[23] Zeynep Pehlivan. On the pursuit of fake news: Graph neural network meets nlp. 2021.

[24] Ferdinand Schaal and Jesper Phillips. Using a word analysis method and gnns to classify misinformation related to 5g-conspiracy and the covid-19 pandemic. volume 2882, 2020.

[25] Sanchit Sinha. Using graph convolutional neural networks for nlp tasks. 2020.

[26] Aolan Sun, Jianzong Wang, Ning Cheng, Huayi Peng, Zhen Zeng, and Jing Xiao. Graphtts: Graph-to-sequence modelling in neural text-to-speech. In *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6719–6723, 2020.

[27] Haitian Sun, Tania Bedrax-Weiss, and William W. Cohen. Pullnet: Open domain question answering with iterative retrieval on knowledge bases and text, 2019.

[28] Ming Tu, Jing Huang, Xiaodong He, and Bowen Zhou. Graph sequential network for reasoning over sequences, 2020.

[29] Shikhar Vashishth. Neural graph embedding methods for natural language processing, 2019.

[30] Xiaoyan Wang, Pavan Kapanipathi, Ryan Musa, Mo Yu, Kartik Talamadupula, Ibrahim Abdelaziz, Maria Chang, Achille Fokoue, Bassem Makni, Nicholas Mattei, and Michael Witbrock. Improving natural language inference using external knowledge in the science questions domain, 2018.

[31] Xinde Wei, Hai Huang, Longxuan Ma, Ze Yang, and Liutong Xu. Recurrent graph neural networks for text classification. In *2020 IEEE 11th International Conference on Software Engineering and Service Science (ICSESS)*, pages 91–97, 2020.

[32] Fan Wu and Xiang Bai. Insertgnn: Can graph neural networks outperform humans in toefl sentence insertion problem? *ArXiv*, abs/2103.15066, 2021.

[33] Lingfei Wu, Yu Chen, Kai Shen, Xiaojie Guo, Hanning Gao, Shucheng Li, Jian Pei, and Bo Long. Graph neural networks for natural language processing: A survey, 2021.

[34] Xinyu Wu, Zheng Luo, Zhanwei Du, Jiaxin Wang, Chao Gao, and Xianghua Li. Twtgnn: Two windows graph-based model for text classification. In *2021 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8, 2021.

[35] Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and Philip S. Yu. A comprehensive survey on graph neural networks. *IEEE Transactions on Neural Networks and Learning Systems*, 32(1):4–24, 2021.

[36] Michihiro Yasunaga, Hongyu Ren, Antoine Bosselut, Percy Liang, and Jure Leskovec. Qa-gnn: Reasoning with language models and knowledge graphs for question answering, 2021.

[37] Michihiro Yasunaga, Rui Zhang, Kshitijh Meelu, Ayush Pareek, Krishnan Srinivasan, and Dragomir Radev. Graph-based neural multi-document summarization, 2017.

[38] Haopeng Zhang and Jiawei Zhang. Text graph transformer for document classification. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8322–8327, Online, November 2020. Association for Computational Linguistics.

[39] Yue Zhang, Qi Liu, and Linfeng Song. Sentence-state lstm for text representation, 2018.

[40] Yufeng Zhang, Xueli Yu, Zeyu Cui, Shu Wu, Zhongzhen Wen, and Liang Wang. Every document owns its structure: Inductive text classification via graph neural networks, 2020.

[41] Ke Zhao, Lan Huang, Rui Song, Qiang Shen, and Hao Xu. A sequential graph neural network for short text classification. *Algorithms*, 14(12):352, 2021.

[42] Jie Zhou, Ganqu Cui, Shengding Hu, Zhengyan Zhang, Cheng Yang, Zhiyuan Liu, Lifeng Wang, Changcheng Li, and Maosong Sun. Graph neural networks: A review of methods and applications. *AI Open*, 1:57–81, 2020.