Improving Natural Language Inference Using External Knowledge in the Science Questions Domain

Under Review

Abstract

Natural Language Inference (NLI) is fundamental to many Natural Language Processing (NLP) applications including semantic search and question answering. The NLI problem has gained significant attention thanks to the release of large scale, challenging datasets. Present approaches to the problem largely focus on learning-based methods that use only textual information in order to classify whether a given premise entails, contradicts, or is neutral with respect to a given hypothesis. Surprisingly, the use of methods based on structured knowledge – a central topic in artificial intelligence – has not received much attention vis-a-vis the NLI problem. While there are many open knowledge bases that contain various types of reasoning information, their use for NLI has not been well explored. To address this, we present a combination of techniques that harness knowledge graphs to improve performance on the NLI problem in the science questions domain. We present the results of applying our techniques on text, graph, and text-to-graph based models, and discuss implications for the use of external knowledge in solving the NLI problem. Our model achieves the new state-of-the-art performance on the NLI problem over the SciTail science questions dataset.

1 Introduction

Natural Language Inference (NLI) – also known as textual entailment – is a fundamental task in Natural Language Understanding (NLU) (MacCartney and Manning 2009). Progress on the NLI problem has been shown to improve performance on important tasks that require NLU, including semantic search, question answering (QA), and text summarization. The main goal in the NLI problem is to determine whether a given natural language *hypothesis* `h` can be inferred from a natural language *premise* `p`. Specifically, NLI is often cast as a classification problem: given two sentences – hypothesis and premise – the problem lies in classifying the relationship between them into one of three classes: 'entailment', 'contradiction', or 'neutral'. Recently, in order to support training of supervised classifiers for NLI, multiple large-scale datasets have been released (Bowman et al. 2015; Khot, Sabharwal, and Clark 2018; Williams, Nangia, and Bowman 2018). The task has thus gained significant attention in the NLP community (Yin, Roth, and Schütze 2018; Parikh et al. 2016; Khot, Sabharwal, and Clark 2018; Chen et al. 2018) and has been specifically identified as a way to increase the accuracy of question answering systems (Boratko et al. 2018). In this paper, we restrict our focus to the "entailment" class, as it is the most salient to down-stream tasks such as question answering. Specifically, we develop a framework that accurately assesses whether a given premise `p` entails a given hypothesis `h`.

Usable inference and reasoning methods have been a central contribution of artificial intelligence (AI) research. Specifically, reasoning methods and the knowledge bases that support them have played an important role in addressing formal reasoning tasks. While the introduction of large datasets for NLI has enabled research into applying learning algorithms by casting NLI as a classification problem, the potential of pre-existing knowledge has been only minimally explored (Chen et al. 2018; Marelli et al. 2014). This knowledge may manifest itself in various forms: as facts consisting of entities and their relationships; as lexical knowledge about the synonyms of entities; or as common-sense and background statements and relationships. Exploiting additional knowledge from different knowledge graphs, as in the present work, or from more sophisticated knowledge bases, may well aid attempts to solve the NLI problem.

Figure 1 shows an example of a subgraph that can be derived from ConceptNet based on the concepts mentioned in a given premise and hypothesis[1]. This subgraph includes connections between the concepts mentioned in the premise and the hypothesis, via concepts available in ConceptNet. Such

[1]The example is based on the SNLI dataset (Bowman et al. 2015)

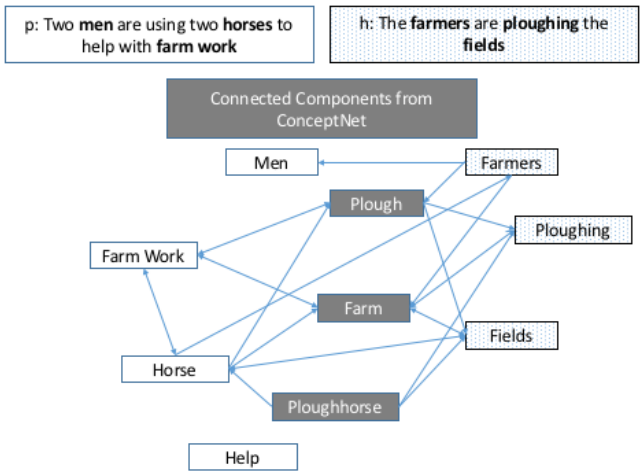

Figure 1: Example of a subgraph from ConceptNet for a given premise `p` and a hypothesis `h`. Edges represent the existence of a relationship between concepts; nodes (concepts) with a gray background are those not mentioned explicitly in either `p` or `h`.

subgraphs can provide additional information to improve the performance of learning-based systems for the NLI problem.

Contributions: In this paper, we introduce the **`ConSeqNet`** framework, which enables the use of various kinds of knowledge bases to retrieve knowledge relevant to a given instance of the NLI problem, by retrieving information related to each premise and hypothesis. We describe the architecture of this framework, and demonstrate its use with a specific external knowledge source – ConceptNet – and evaluate its performance on three different sources (ConceptNet, Wordnet, DBpedia). We compare the performance of three distinct approaches to augmenting the knowledge used to train for and to predict entailment relationships between given pairs of premises and hypotheses: graph-only, text-only, and text-and-graph. Using both qualitative and quantitative results, we demonstrate that introducing graph-based features boosts performance on the NLI problem, but only when text features are present as well. Our system establishes a new state-of-the-art with a test set performance (accuracy) of 85.2% on the SciTail dataset. We conclude with some ideas for future work that harness our general framework.

2 Background and Related Work

We first consider prior work in the two main areas explored in this paper: the Natural Language Inference (NLI) problem, and Knowledge Graphs; as well as work at their intersection.

2.1 Natural Language Inference

Recently, the NLI task has gained significant attention due to the release of multiple crowd-sourced, large-scale datasets that can be used to train neural network classifiers (Yin, Roth, and Schütze 2018; Parikh et al. 2016; Khot, Sabharwal, and Clark 2018; Chen et al. 2018). In particular, the SNLI (Bowman et al. 2015) dataset[2] has been a major catalyst of research on NLI facilitating learning-based approaches by providing a sufficient number of training examples for data-intensive learning algorithms. The Multi Genre NLI corpus (MultiNLI) (Williams, Nangia, and Bowman 2018) is an effort to address the limitations SNLI. The dataset focuses on domain adaptation by introducing genre labels for each sentence pair. SciTail (Khot, Sabharwal, and Clark 2018), the primary focus of which is the down-stream task of standardized-test question-answering. It was semi-automatically created from the ARC dataset (Clark et al. 2016) released by AI2. In this work, we primarily focus on SciTail dataset, which allows us to address the problem of Question Answering using an entailment system in our future work.

Neural networks with an encoder-attention-classifier architecture are commonly used for NLI tasks (Bowman et al. 2015; Wang, Hamza, and Florian 2017; Khot, Sabharwal, and Clark 2018). To encode the premise and hypothesis, most methods use RNNs, which are widely used for many NLP tasks. Bowman et al. (2015) used LSTMs to learn sentence representations for premise and hypothesis separately, concatenating them for classification. Rocktäschel et al. (2015) introduced a word-by-word attention model that learns conditional encodings of premise and hypothesis for textual entailment. The match-LSTM model (Wang and Jiang 2015) extended the model presented by Rocktäschel et al. (2015) to address the limitation of a single vector representation of the premise for learning attention weights. The next step in this direction was the multi-perspective matching mechanism introduced by Wang, Hamza, and Florian (2017). Most of these text-based models differ from each other in their attention mechanisms. While the above-mentioned approaches used word-by-word attention mechanisms, Yin, Roth, and Schütze (2018) has explored inter-sentence interactions for textual entailment, leading to the current state-of-the-art performance on the SciTail dataset; however, other NLI datasets were not used for evaluation.

2.2 External Knowledge (Knowledge Graphs)

Most openly available external knowledge[3] sources are in the form of graphs, commonly known as Knowledge Graphs (KG). Some of the well known KGs include Freebase, DBpedia, Yago, and ConceptNet. A KG is defined as a set of concepts connected by relationships where the concepts form the nodes of the graph and the relationships are the labeled edges. A fact such as "Barack Obama is the spouse of Michelle Obama" is represented in a KG (for example on DBpedia) as `dbr:Barack_Obama dbo:spouse dbr:Michelle_Obama`, where `dbr:Barack_Obama` and `dbr:Michelle_Obama` are nodes and `dbo:spouse` is a labeled edge in DBpedia.

KGs have been used extensively in Information Retrieval (Bouchoucha, Liu, and Nie 2014; Xiong and Callan 2015), Recommendation Systems (Lalithsena et al. 2017;

[2] https://nlp.stanford.edu/projects/snli/

[3] We use External Knowledge and Knowledge Graphs interchangeably in this paper.

Kapanipathi et al. 2014), and Natural Langugage Processing (Hoffart et al. 2012). The prominent concern when using knowledge graphs is the availability of relevant knowledge, both in terms of applicability to the task setting and to the domain/topic of interest. This is closely related to the way in which these knowledge graphs are created. For instance, DBpedia is a generic knowledge base with information extracted from Wikipedia infoboxes; these contain facts such as (*Barack_Obama, spouseof, Michelle_Obama*). By contrast, ConceptNet (Liu and Singh 2004) consists of common-sense knowledge acquired through crowd sourcing. While DBpedia is well suited for entity-based tasks such as movie recommendation and entity disambiguation, ConceptNet may be more appropriate if common sense reasoning is required. WordNet is a lexical database offering of synsets (word-sense synonym sets) connected by a small number of semantic relationships such as hypernym, hyponym, and antonyms. In this work, we experiment with the three above mentioned KGs, i.e., DBpedia, ConceptNet, and Wordnet.

2.3 Knowledge Graphs and NLI

Few prior approaches have exploited syntactic structure for textual entailment. The Decomposed Graph Entailment Model (Khot, Sabharwal, and Clark 2018) (DGEM) – which is an extension to the Decomposable Attention model (Parikh et al. 2016) (DecompAtt) – uses OpenIE to construct syntactic structures (as graphs) for hypothesis, and computes node attention based on the premise text. While graph structures represent the existing hypothesis, external knowledge sources are not used in this model to enhance the textual content of either the premise or the hypothesis. One work which closely relates to the objective of this paper is that of Chen et al. (2017). This work primarily focuses on WordNet as the external knowledge that is exploited for NLI. WordNet, however, is a lexical database that is restricted to a small number of linguistic relationships among terms. Furthermore, Chen et al. (2017) uses only four relationships in order to generate features based on WordNet. In our work, we do not restrict the external knowledge source to a lexical database such as WordNet. We also explore more expressive knowledge graphs such as DBpedia and ConceptNet. Also, the most important difference is the way we use KGs: our method enhances the texts (as graphs) and builds matching model over the enhanced texts for textual entailment.

3 Approach

In this section, we present the architecture of our system called **`ConSeqNet`**, and the approach underlying it. Figure 2 depicts the framework of **`ConSeqNet`** – the system can use both textual as well as structural information from knowledge graphs to assist in determining textual entailment. The framework can be clearly divided into two parts: (a) a **text based model** that takes in as input the premise and hypothesis text; and (2) a **graph based model** whose input is specific knowledge derived from the knowledge base using the given premise and hypothesis. We present the details of these models in turn.

3.1 Text Based Model

Most models developed for NLI have relied on the text of the given premise and hypothesis, without the aid of any external knowledge. These models follow an encode-attend-classify approach. First, the premise and hypothesis are encoded using recurrent neural networks (RNNs). Next, an attention layer is implemented on top of the encoders. The final layer is then used for classification. In our work, we have primarily used match-LSTM as our text based model. We opted for match-LSTM for two main reasons: (1) match-LSTM has been a critical component of multiple reading comprehension techniques (Wang and Jiang 2015; 2016); and (2) our implementation of match-LSTM performed significantly better than the baselines (details in Section 4).

Consider a premise $P = (t_1^p, t_2^p, ..., t_K^p)$ and a hypothesis $H = (t_1^h, t_2^h, ..., t_N^h)$, where t_i^p and t_j^h are embedding vectors of the corresponding words in premise and hypothesis. The model then computes the matching results between P and H as follows:

- **Context Encoding:** A contextual representation of premise and hypothesis is generated by encoding their embedding vectors using BiLSTMs. Let $\mathbf{p}_i$ and $\mathbf{h}_j$ be the contextual representation of the i-th word in the premise and the j-th word in the hypothesis.
- **Word-by-Word Attention:** This layer computes the inter-attention between the contextual embeddings of the premise and hypothesis. The entries of the (unnormalized) attention matrix $\mathbf{E} \in \mathbb{R}^{K \times J}$ are defined as:

$$E_{ij} = \mathbf{p}_i \cdot \mathbf{h}_j \tag{1}$$

  we can then compute the soft alignment α_j for the hypothesis as follows:

$$\alpha_j = \sum_{i=1}^{K} \frac{\exp(E_{ij})}{\sum_{k=1}^{J} \exp(E_{kj})} \mathbf{p}_i \tag{2}$$

- **Matcher:** We compare the soft aligned premise and hypothesis at each word position as a feature vector:

$$\tilde{\mathbf{h}}_j = [\mathbf{h}_j; \alpha_j; \mathbf{h}_j - \alpha_j; \mathbf{h}_j \odot \alpha_j],$$

  where $\odot$ denotes the element-wise multiplication, and $[;]$ denotes vector concatenation. The feature vectors for all the hypothesis positions are fed into a BiLSTM to get the matching states $\{\mathbf{h_j^m}\}_{j=1:N}$.
- **Pooling:** To get a fixed-sized vector representation for the matching result, we apply max-pooling across $\{\mathbf{h_j^m}\}_{j=1:N}$:

$$\mathbf{x}_{\text{out}}^{\text{text}} = \mathbf{h}_{\text{max}}^{\mathbf{m}} = \max([\mathbf{h_1^m}, \mathbf{h_2^m}, \ldots, \mathbf{h_N^m}]) \tag{3}$$

  $\mathbf{x}_{\text{out}}^{\text{text}}$ is then used for classification.

3.2 Graph Based Model

The input to the graph based model is a premise graph and a hypothesis graph respectively. As shown in Figure 2,

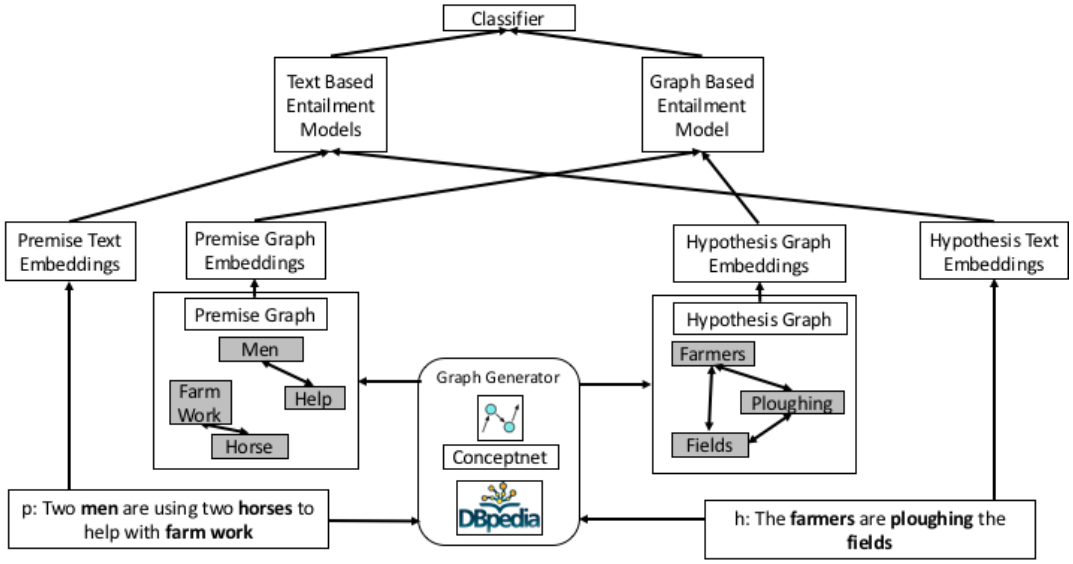

Figure 2: The overall architecture of the **ConSeqNet** system, illustrated via an example.

the first step is to transform the given premise and hypothesis text into a relevant subgraph mapped to an external knowledge. Given premise $P = (t_1^p, t_2^p, ..., t_K^p)$ and hypothesis $H = (t_1^h, t_2^h, ..., t_M^h)$, the transformation generates $P_g = (V_p, E_p)$ and $H_g = (V_h, E_h)$ where $V_p = (v_1^p, ..., v_K^p)$ and $V_h = (v_1^h, ..., v_J^h)$ are a subset of concepts in the knowledge graph. E_p and E_h are labeled edges connecting concepts in P_g and H_g respectively. H_g and P_g are the input for our graph-based entailment model.

Premise and Hypothesis Graphs Premise and hypothesis graphs are generated by mapping text phrases to concepts in the external knowledge graph. For instance, as shown in Figure 1, given hypothesis *"The farmers are ploughing the fields"*, we map "*farmers*", "*ploughing*, and "*fields*" in the sentence to their corresponding concepts in the external knowledge. We use the following techniques to generate three different types of subgraphs for each premise and each hypothesis separately:

- *Concepts Only*[4]*:* For each premise and hypothesis, we use lexical mappings within each knowledge source to map individual words and phrases to concepts. Those concepts make up the vertices of the *Concepts Only* subgraph. All edges connecting those vertices are also included in the subgraph.
- *One-Hop:* The concepts in the *Concepts Only* graph for premise and hypothesis are expanded to include all their one hop neighbors in the external knowledge source.
- *Two-Hop:* Two-Hop graphs were generated by adding two hop neighbors to the *Concepts Only* graph, but only if they constitute paths between vertices that were already in the *Concepts Only* graph. This can be thought of as a graph that increases interconnectedness without introducing new concepts. We took this approach because we observed that *One-Hop* graphs turned to be noisy (on average, the *One-Hop* graphs generated using ConceptNet increased the number of concepts from 9 to 326).

We use the graphs constructed (under the three conditions above) for premise and hypothesis as the input to our neural network model for entailment. In this work, we explore two different neural network models: (1) match-LSTM; and (2) GDecomp Attention Model.

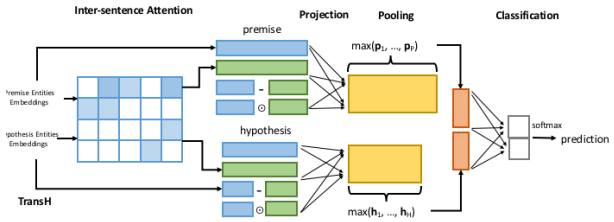

Figure 3: The graph model in **ConSeqNet**, which is used to cast external knowledge into the form of a graph.

Gmatch-LSTM for Concept-Only Graphs Our first design choice was to use match-LSTM with the *Concepts Only* graphs P_g and H_g. The key idea is to treat each node (concept) in the vertex set of a given graph (V_p or V_h) as a single token, and re-arrange them into a sequence of concepts. This sequence is ordered by the positions in the original texts that the concepts respectively align to. Thus, match-LSTM can be applied to match the sequence of premise concepts and the sequence of hypothesis concepts.

[4]Concepts in the *Concepts Only* graph are used to form the base set for *One-Hop* and *Two-Hop* graph generation techniques.

In contrast with the text based model, where each word is first represented as its word embedding vector, in the graph based match-LSTM each token (concept) is initialized using the corresponding **Concept Embedding**. The concept embeddings are trained by knowledge graph embedding techniques such as TransH (Wang et al. 2014), and ComplEX (Trouillon et al. 2016).

GDecomp Attention Model for General Graphs The use of match-LSTM is justified when the input is the *Concepts Only* graph, since the concept can be aligned in the order of their appearance in premise and hypothesis to form text sequences. However, it is non-trivial to order the concepts in the graphs generated by the *One Hop* and *Two Hop* strategies; this makes the use of match-LSTM unintuitive. We therefore develop a new model *GDecomp Attention* to overcome this limitation.

The overall schematic of the graph based model is depicted in Figure 3. The embeddings of premise and hypothesis concepts from the graph are the input to the neural network. An attention weighted representation of both premise and hypothesis is determined, and is transformed to a final fixed size representation using pooling. This representation of premise and hypothesis (concatenated) is used for classification. Below, we distinguish the graph based model from the match-LSTM (text based) model described previously:

- **Context Encoding:** Since the concept graphs may not have sequential structures, the GDecomp model directly starts with the **concept embeddings**.
- **Word-by-Word Attention:** This layer computes the inter-attention between the embeddings of the concepts in premise and hypothesis to find the best aligned concepts between the respective graphs. The major difference from match-LSTM is that GDecomp performs two-way attention (Seo et al. 2016). The model computes the soft alignment for premise and hypothesis respectively as follows:

$$\beta_i = \sum_{j=1}^{J} \frac{\exp(E_{ij})}{\sum_{k=1}^{K}\exp(E_{ik})}\mathbf{h}_j, \quad \alpha_j = \sum_{i=1}^{K} \frac{\exp(E_{ij})}{\sum_{k=1}^{J}\exp(E_{kj})}\mathbf{p}_i$$

- **Matcher:** Due to the lack of sequential structure, the matching nodes of each hypothesis concept j or premise concept *is* are computed with a projection feed-forward network:

$$\mathbf{p_i^m} = F\left([\mathbf{p}_i; \beta_i; \mathbf{p}_i - \beta_i; \mathbf{p}_i \odot \beta_i]\right)$$
$$\mathbf{h_j^m} = F\left([\mathbf{h}_j; \alpha_j; \mathbf{h}_j - \alpha_j; \mathbf{h}_j \odot \alpha_j]\right)$$

  where $F(\cdot)$ denotes a feed-forward network; $\odot$ denotes the element-wise multiplication; and $[;]$ denotes vector concatenation.

- **Pooling:** Finally, we apply max-pooling and mean-pooling across all matching nodes in the premise and hypothesis. The output of pooling is concatenated, and is presented as the output of the graph model, which can then be used to make the final prediction. $\mathbf{s} \in \{\mathbf{h}, \mathbf{p}\}$.

$$\mathbf{s}'_{\text{max}} = \max\left([\mathbf{s_1^m}, \mathbf{s_2^m}, \ldots, \mathbf{s_N^m}]\right) \tag{4}$$

$$\mathbf{s}'_{\text{avg}} = \text{avg}\left([\mathbf{s_1^m}, \mathbf{s_2^m}, \ldots, \mathbf{s_N^m}]\right)$$

$$\mathbf{x}^{\text{graph}}_{\text{out}} = \left[\mathbf{p}'_{\text{max}}; \mathbf{p}'_{\text{avg}}; \mathbf{h}'_{\text{max}}; \mathbf{h}'_{\text{avg}}\right]$$

3.3 Merging Text and Graph Models

The final results of matching the text model $\mathbf{x}^{\text{text}}_{\text{out}}$ and the graph model $\mathbf{x}^{\text{graph}}_{\text{out}}$ 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:

$$\mathbf{pred} = F\left(\left[\mathbf{x}^{\text{text}}_{\text{out}}; \mathbf{x}^{\text{graph}}_{\text{out}}\right]\right) \tag{5}$$

It is important to note that the final hidden state of the graph model can be directly fed into a classifier. This can be considered an entailment model that uses only information from external knowledge. We have used this model (without textual information) to make decisions on selecting the graph construction technique for our approach (more details in Section 4.3).

4 Experiments and Results

In this section, we detail the experiments that we perform to evaluate our **`ConSeqNet`** system. We first describe each dataset, followed by the training setup of our best model and implementation details. We then compare that model's performance to numbers from recent entailment models and baselines. In the latter parts of this section, we discuss the selection of knowledge graphs and the performance of graph construction techniques – specifically, the *Concepts Only, One Hop,* and *Two Hop* methods (c.f. Section 3).

4.1 Datasets

We use the SciTail corpus (Khot, Sabharwal, and Clark 2018), which is a textual entailment dataset derived from the AI2 Reasoning Challenge (ARC) multiple choice question answering corpus (Clark and Gardner 2017). The dataset contains 27,026 sentence pairs (premise and hypothesis), with binary labels denoting whether the relationship between each pair is `entails` or `neutral`. The hypothesis is created using questions that contain correct answers, from the multiple answer options; and the premise is retrieved from the ARC corpus[5]. The performance of the entailment models on this dataset is shown in Table 1; we use accuracy as our evaluation metric.

4.2 Training Setup

All words in the text model are initialized by 300D Glove vectors (glove.840B.300d.zip)[6], and the concepts that act as the input for the graph model are initialized by 300D ConceptNet PPMI vectors (Speer, Chin, and Havasi 2017); these are openly available for ConceptNet. We use the pre-trained embeddings without any fine tuning. We have adapted match-LSTM with GRUs as our text and graph based model. The system is trained by Adagraph with a

[5] http://data.allenai.org/arc/arc-corpus/

[6] https://nlp.stanford.edu/projects/glove/

Model	Dev	Test
Decomp-Attn (Parikh et al. 2016)	75.4	72.3
DGEM (Khot, Sabharwal, and Clark 2018)	79.6	77.3
DeIsTe (Yin, Roth, and Schütze 2018)	82.4	82.1
BiLSTM-Maxout (Mihaylov et al. 2018)	-	84.0
match-LSTM (Wang and Jiang 2015)	88.2	84.1
Our implementation		
match-LSTM (GRU)	88.5	84.2
match-LSTM – Wordnet (Chen et al. 2018)	88.8	84.3
match-LSTM + Gmatch-LSTM (**ConSeqNet**)	**89.6**	**85.2**

Table 1: Performance of entailment models on SciTail in comparison to our best model that uses match-LSTM as the text and the graph model with *Concept-Only graph* and CN-PPMI embeddings.

learning rate of 0.001, and batch size of 40. Both the text and graph based models are trained jointly. For the graph model, we use concepts from the *Concepts Only* graph, which is generated using the approach detailed in Section 3.2.

4.3 Implementation

We used the AllenNLP[7] library to implement all the models used in the experiments. While the previous section (Training Setup) specifically focused on our best performing model, here we provide implementation details for all the experiments that were performed. We used a plug-and-play approach where we varied: (a) text models (match-LSTM, DeCompAttn); (b) graph models (match-LSTM, GDecompAttn); (c) external knowledge sources (DBpedia, WordNet, and ConceptNet); (d) graph construction techniques (*Concept-Only, One Hop,* and *Two Hop*); and (e) graph embeddings (CN-PPMI, TransH).

In order to map text to concepts, we used DBpedia Spotlight (Mendes et al. 2011) for DBpedia, and Spacy[8] to implement a max-substring match for Wordnet and ConceptNet. While ConceptNet had openly available embeddings (CC-PPMI), we used TransH embeddings (Wang et al. 2014) generated using OpenKE[9] with default configurations for the other knowledge sources. We plan to open source our code, as well as our modified version of the SciTail dataset which includes annotations from ConceptNet and DBpedia.

4.4 Baselines and Comparison

We compare our work to the following baselines: (1) Decomposable Attention Model (Decomp-Att) (Parikh et al. 2016); (2) Decomposed Graph Entailment Model (DGEM) (Khot, Sabharwal, and Clark 2018), which is the first of the entailment models to use graph structure from OpenIE as external knowledge and show improvements on SciTail; (3) Deep explorations of Inter-sentence interactions for Textual entailment (DeIsTe) (Yin, Roth, and Schütze 2018), which presently has the state of the art performance on the SciTail dataset (Yin, Roth, and Schütze 2018); (4) BiLSTM-Maxout (Mihaylov et al. 2018), the latest entailment model that has shown promise on the SciTail dataset[10]; (4) match-LSTM (Wang et al. 2015), which has shown good performance on the SNLI dataset; (5) match-LSTM with GRU, which replaces LSTMs with GRUs for its encoding, since GRUs are better than LSTMs for text classification tasks (Yin et al. 2017) (this is the match-LSTM model used in our **ConSeqNet** model); and (5) match-LSTM – Wordnet features Chen et al. (2018), which uses five features based on synonyms, antonyms, hypernyms, and co-hypernyms from Wordnet (external knowledge) in its co-attention mechanism to improve performance on SNLI. We reimplement these five features and add them to baseline (5).

Table 1 shows the performance of our **ConSeqNet** model in comparison to other entailment models. Two of the five models in Table 1 use some kind of external knowledge. Almost all the match-LSTM variations exhibit accuracy greater than 84%, which is better than the recent entailment models on which the SciTail dataset has been tested. Similar to the results shown in (Yin et al. 2017), using GRUs with match-LSTM slightly improved the accuracy (particularly on the dev set). The accuracy of our jointly trained model **ConSeqNet** is 89.6% on the dev set and 85.2% on the test set. Later in this paper, we expound further on our model – this includes choices such as the use of different knowledge sources, graph models, and the graph construction technique.

	DBpedia	Wordnet	ConceptNet
Entities/Concepts	5M	155K	1.1M
Relationships	1100	16	40
Facts	33M	117K	3.15M

Table 2: Comparison between different knowledge sources based on the number of entities, relationships, and facts.

4.5 Selecting External Knowledge Source

In this work, we focus on openly available knowledge graphs. Based on the knowledge graphs' availability for use and their distinct properties, we chose DBpedia, ConceptNet, and WordNet. In Table 2, we provide details on each of these knowledge graphs. DBpedia is the largest with more than 5 million entities and 33 million facts. ConceptNet subsumes WordNet conceptually, and both contain general type information; however, the reliability of WordNet's linguistic features is higher than ConceptNet's.

For the NLI problem, Wordnet has only been partially explored. Wordnet is a lexical database with a restricted set of relationships between terms. Chen et al. (2018) use Wordnet by creating five new knowledge-based features that are added (for co-attention) to their model. The five features are derived from synonym/antonym and hypernym/hyponym relationships in Wordnet between terms in the premise and hypothesis. In Table 1, we show that using the expanded set of

[7]https://allennlp.org/

[8]https://spacy.io/

[9]http://openke.thunlp.org/

[10]The details of the BiLSTM-Maxout (Mihaylov et al. 2018) model are not available. We have reported the numbers provided in a pre-print of a publication that is to appear.

features from WordNet – used in the attention mechanism of match-LSTM – has an accuracy of 84.3% on the SciTail datset. While Wordnet is useful, it is restrictive in terms of its applicability, particularly because of its coverage and the types of relationships available. DBpedia and ConceptNet are larger knowledge sources, as shown in Table 2.

In terms of the number of concepts mapped from text to external knowledge, ConceptNet has more concepts (9) on average in comparison to DBpedia (6). While it is important that the graphs are not noisy, it is also important to have enough information to exploit. In which case, ConceptNet is slightly better with few more concepts per sentence for the SciTail dataset.

The types of relationships between concepts in DBpedia are factual and derived from Wikipedia. Based on qualitative analysis of these relationships, we found that they may not be suitable for NLI datasets, and specifically SciTail. On the other hand, we found that, ConceptNet with its 40 relationships expressing common sense knowledge may be more suitable. In order to quantitatively select the right knowledge source, we determined the impact of the each knowledge graph on SciTail. We created the *Concept-Only* graph from each of the available external knowledge sources, and evaluated them using our match-LSTM-GDecomp model on the dev set of SciTail. The results of this experiment are shown in Table 3; based on these results, we decided to use ConceptNet for all our graph-based experiments (detailed next).

Knowledge Sources	Accuracy
Wordnet	87.6
DBpedia	87.3
ConceptNet	**88.6**

Table 3: Using Knowledge Sources for SciTail: Results. The model used for these experiments is match-LSTM+GDecomp

4.6 Graph Generation Experiments

In order to select the graph generation mechanism (c.f. Section 3.2) that has the highest impact on the NLI problem, we ran experiments with match-LSTM as our text model and GDecomp as our graph model. GDecomp was chosen because the concepts retrieved from *One-Hop* and *Two-Hop* do not have any specific sequence or ordering. Table 4 presents the results of these experiments. The hyperparameters remained the same for all three graph generation experimental conditions.

All the graph + text models perform equally well. However, when only the graph model is considered, the *One-Hop* graph exhibits lower accuracy in comparison to the *Concept Only* and *Two-Hop* graphs. This may be due to the noise induced from external knowledge and the addition of a large number of concepts in the *One-Hop* case. On average, premise and hypothesis sentences consist of 19 and 12 words respectively, but their respective *One-Hop* graphs have over 300 concepts, on average. On the other hand, the *Concept Only* graphs average 9 and 6 concepts, and they perform better than the *One-Hop* graph model (72.3% vs 68.2%). Based on these results, and the simplicity of the graph construction technique in terms of the number of concepts and features input to our graph model, we decided to focus our exploration on the *Concept Only* graph representation.

The accuracy of the graph only models are relatively low, whereas the graph+text and text only models are comparable. This might lead to the conclusion that the graph+text model is only driven by the text model. However, an *Oracle* model condition, where we choose the correct answer between the Text and Graph Models, indicates that the Graph model contributes to improved accuracy on the dev set (88.5 for text only from Table 1 versus at least 91.6 for Oracle). With *One-Hop*, the graph model gets almost 40% of the answers correct that are incorrectly predicted by the text-only model. With this we can conclude that there is value to using external knowledge for Natural Language inference on SciTail. This is contrary to recent results by Mihaylov et al. (2018), who conclude that external knowledge such as ConceptNet is not useful in this domain.

Graph Generation	Graph Model Accuracy	Graph + Text Model Accuracy	Oracle Text ∨ Graph	Avg Concepts Premise (19 words)	Avg Concepts Hypothesis (12 words)
Concepts	72.3	87.2	**92.5**	9	6
One Hop	68.2	87.3	**93.1**	369	312
Two Hop	71.7	87.3	**91.6**	23	15

Table 4: Performance of graph generation techniques on the match-LSTM++ GDecomp Model. The accuracy of the models are on the SciTail dev set.

4.7 Selecting Text+Graph Model

We experimented with many text models and selected match-LSTM due to its superior performance on SciTail. In order to determine the best combination of the graph models (Gmatch-LSTM and GDecomp) with match-LSTM as the text model, we ran multiple experiments on the SciTail Dev set. The results are shown in Table 5. Both GDecomp and Gmatch-LSTM are competitive in their performance. We also experimented with different knowledge graph embeddings to determine their impact on these models. ConceptNet-CN-PPMI clearly shows an improvement **(89.6)** in comparison to other models on the Dev set. This model is used as our final model to be evaluated and compared against the baselines, leading to new state of the art performance numbers as shown in Table 1.

Text Model	Graph Model	Embedding	Dev
match-LSTM	Gmatch-LSTM	ConceptNet-CN-PPMI	**89.6**
match-LSTM	Gmatch-LSTM	ConceptNet-TransH	87.3
match-LSTM	Gmatch-LSTM	ConceptNet-CompleX	88.3
match-LSTM	GDecomp	ConceptNet-CN-PPMI	88.6
match-LSTM	GDecomp	ConceptNet-TransH	87.6
match-LSTM	GDecomp	ConceptNet-CompleX	88.4

Table 5: Results of all combinations of Text and Graph Models along with various ways of computing embeddings on the SciTail dataset.

5 Conclusion & Future Work

In this paper, we presented the **ConSeqNet** system, which is an entailment model for solving the Natural Language Inference (NLI) problem that utilizes ConceptNet as an external knowledge source. Our model has the current state-of-the-art performance on the NLI classification problem, with an accuracy of 85.2% on the SciTail dataset. We analyze various external knowledge sources and their effect on the NLI problem, and show that there is promise in using knowledge graphs such as ConceptNet for textual entailment. This is in direct contrast to other studies that have found that external knowledge does not improve performance on existing models for textual entailment.

Our future work includes designing a framework to exploit multiple relevant knowledge sources based on the given dataset and context. Existing external knowledge sources are known to be extremely noisy, and new techniques must be developed in order to extract knowledge relevant to a specific task (such as NLI). Another interesting direction involves exploring new ways to represent the structure of premise and hypothesis subgraphs, and systematically using the relations between the concepts contained therein to improve performance on the NLI task.

References

Boratko, M.; Padigela, H.; Mikkilineni, D.; Yuvraj, P.; Das, R.; McCallum, A.; Chang, M.; Fokoue-Nkoutche, A.; Kapanipathi, P.; Mattei, N.; Musa, R.; Talamadupula, K.; and Witbrock, M. 2018. A systematic classification of knowledge, reasoning, and context within the arc dataset. In *Proceedings of the Machine Reading for Question Answering (MRQA) Workshop at Annual Meeting of the Association for Computational Linguistics (ACL)*.

Bouchoucha, A.; Liu, X.; and Nie, J.-Y. 2014. Integrating multiple resources for diversified query expansion. In de Rijke, M.; Kenter, T.; de Vries, A. P.; Zhai, C.; de Jong, F.; Radinsky, K.; and Hofmann, K., eds., *Advances in Information Retrieval*, 437–442.

Bowman, S. R.; Angeli, G.; Potts, C.; and Manning, C. D. 2015. A large annotated corpus for learning natural language inference. *arXiv preprint arXiv:1508.05326*.

Chen, D.; Fisch, A.; Weston, J.; and Bordes, A. 2017. Reading wikipedia to answer open-domain questions. *arXiv preprint arXiv:1704.00051*.

Chen, Q.; Zhu, X.; Ling, Z.-H.; Inkpen, D.; and Wei, S. 2018. Neural natural language inference models enhanced with external knowledge. In *Proceedings of ACL 2018*.

Clark, C., and Gardner, M. 2017. Simple and effective multi-paragraph reading comprehension. *arXiv preprint arXiv:1710.10723*.

Clark, P.; Etzioni, O.; Khot, T.; Sabharwal, A.; Tafjord, O.; Turney, P. D.; and Khashabi, D. 2016. Combining retrieval, statistics, and inference to answer elementary science questions. In *Proceedings of the 30th AAAI Conference on Artificial Intelligence (AAAI)*, 2580–2586.

Hoffart, J.; Seufert, S.; Nguyen, D. B.; Theobald, M.; and Weikum, G. 2012. Kore: keyphrase overlap relatedness for entity disambiguation. In *Proceedings of the 21st ACM international conference on Information and knowledge management*, 545–554.

Kapanipathi, P.; Jain, P.; Venkataramani, C.; and Sheth, A. 2014. User interests identification on twitter using a hierarchical knowledge base. In *European Semantic Web Conference*, 99–113.

Khot, T.; Sabharwal, A.; and Clark, P. 2018. SciTail: A textual entailment dataset from science question answering. In *Proceedings of the 32nd AAAI Conference on Artificial Intelligence (AAAI)*.

Lalithsena, S.; Perera, S.; Kapanipathi, P.; and Sheth, A. 2017. Domain-specific hierarchical subgraph extraction: A recommendation use case. In *Big Data (Big Data), 2017 IEEE International Conference on*, 666–675.

Liu, H., and Singh, P. 2004. Conceptnet—a practical commonsense reasoning tool-kit. *BT technology journal* 22(4):211–226.

MacCartney, B., and Manning, C. D. 2009. *Natural language inference*. Stanford University Stanford.

Marelli, M.; Bentivogli, L.; Baroni, M.; Bernardi, R.; Menini, S.; and Zamparelli, R. 2014. Semeval-2014 task 1: Evaluation of compositional distributional semantic models on full sentences through semantic relatedness and textual entailment. In *Proceedings of the 8th international workshop on semantic evaluation (SemEval 2014)*, 1–8.

Mendes, P. N.; Jakob, M.; García-Silva, A.; and Bizer, C. 2011. Dbpedia spotlight: shedding light on the web of documents. In *Proceedings of the 7th international conference on semantic systems*, 1–8.

Mihaylov, T.; Clark, P.; Khot, T.; and Sabharwal, A. 2018. Can a suit of armor conduct electricity? A new dataset for open book question answerin. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Parikh, A. P.; Täckström, O.; Das, D.; and Uszkoreit, J. 2016. A decomposable attention model for natural language inference. *arXiv preprint arXiv:1606.01933*.

Rocktäschel, T.; Grefenstette, E.; Hermann, K. M.; Kočiskỳ, T.; and Blunsom, P. 2015. Reasoning about entailment with neural attention. *arXiv preprint arXiv:1509.06664*.

Seo, M.; Kembhavi, A.; Farhadi, A.; and Hajishirzi, H. 2016. Bidirectional attention flow for machine comprehension. *arXiv preprint arXiv:1611.01603*.

Speer, R.; Chin, J.; and Havasi, C. 2017. Conceptnet 5.5: An open multilingual graph of general knowledge. In *AAAI*, 4444–4451.

Trouillon, T.; Welbl, J.; Riedel, S.; Gaussier, É.; and Bouchard, G. 2016. Complex embeddings for simple link prediction. In *International Conference on Machine Learning*, 2071–2080.

Wang, S., and Jiang, J. 2015. Learning natural language inference with lstm. *arXiv preprint arXiv:1512.08849*.

Wang, S., and Jiang, J. 2016. Machine comprehension using match-lstm and answer pointer. *arXiv preprint arXiv:1608.07905*.

Wang, Z.; Zhang, J.; Feng, J.; and Chen, Z. 2014. Knowledge graph embedding by translating on hyperplanes. In *AAAI*, 1112–1119.

Wang, H.; Bansal, M.; Gimpel, K.; and McAllester, D. 2015. Machine comprehension with syntax, frames, and semantics. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, volume 2, 700–706.

Wang, Z.; Hamza, W.; and Florian, R. 2017. Bilateral multi-perspective matching for natural language sentences. *arXiv preprint arXiv:1702.03814*.

Williams, A.; Nangia, N.; and Bowman, S. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, 1112–1122. Association for Computational Linguistics.

Xiong, C., and Callan, J. 2015. Query expansion with freebase. In *Proceedings of the 2015 international conference on the theory of information retrieval*, 111–120.

Yin, W.; Kann, K.; Yu, M.; and Schütze, H. 2017. Comparative study of cnn and rnn for natural language processing. *CoRR* abs/1702.01923.

Yin, W.; Roth, D.; and Schütze, H. 2018. End-task oriented textual entailment via deep explorations of inter-sentence interactions. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, volume 2, 540–545.