ICDM 2019 Knowledge Graph Contest: Team UWA

Michael Stewart
The University of Western Australia
Perth, Australia
michael.stewart@research.uwa.edu.au

Majigsuren Enkhsaikhan
The University of Western Australia
Perth, Australia
majigsuren.enkhsaikhan@research.uwa.edu.au

Wei Liu
The University of Western Australia
Perth, Australia
wei.liu@uwa.edu.au

I. Model

A. Introduction

We begin our report by discussing the challenges we experienced and the motivation behind our approach. We then describe each component of our system in detail.

Our first approach for addressing the contest specification was a novel end-to-end, deep learning-based system. The most challenging task was to find a way to represent the data; considering a sentence may have zero or many triples, and that the relations should be obtained directly from the text, it was exceedingly difficult to represent the input data in such a way that allowed the model to predict a decently-sized set of valid triples from a given document. Our best deep learning-based approach produced high-quality triples, but only in very small numbers. We hence decided to veer away from deep learning and capitalise on the wide variety of readily-available natural language processing tools.

For general English text, resources are available including several annotated benchmark datasets and off-the-shelf tools. For example, CoNLL-2003 English benchmark dataset [1] is a collection of Reuters news-wire articles, annotated with four entity types: persons, organizations, locations, and miscellaneous names[1]. It contains around 300,000 tokens of 22,137 sentences. OntoNotes5.0 [2] is an annotated corpus of 2.9 million words from news, phone conversations, weblogs, broadcast, talk shows in three languages (English, Chinese, and Arabic) with structural information (syntax and predicate argument structure) and shallow semantics (word sense linked to an ontology and coreference)[2]. Off-the-shelf standard named entity recognition (NER) tools are able to recognize named entities of a restricted list of pre-defined entity types, such as location, person names, organization names, money, date, and time. Popular tools include NLTK [3], SpaCy [4], Stanford Named Entity Recogniser [5] and AllenNLP [6], [7].

However, when it comes to real-world applications, such as the domain specific text in automotive engineering or public security, we face the low-resource data problem similar to machine translation between rare languages. There is no benchmark annotated dataset relevant to those domains, and it is near-impossible to find the right pivot language that allows us to take advantages of existing high resource NER tools. In automotive engineering domain, car types and car related names are more important than person or organisation names. For example, in the sentence *Ford re-tuned the suspension and magnetic dampers to allow the GT350 to stiffen the suspension for better performance on the track*, the important entities are *Ford*, *GT350*, *suspension*, and *magnetic dampers*, but NER tools can only capture *Ford* and *GT350* as entities and ignore the other phrases. In order to avoid missing salient information units, chunking of noun phrases for entities and chunking of action related phrases for relations are performed in this work.

Our team also experimented with Open Information Extraction (OpenIE) [8] and knowledge graph construction systems. There are a wide range of OpenIE systems available, with recent approaches incorporating neural networks in order to maximise performance [9]. We found that OpenIE tends to produce a vast number of triples, with many subjects or objects being long sequences of words as opposed to useful entities. This is detrimental to the contest task, which demands a refined set of high-quality triples. Knowledge graph construction systems, such as T2KG [10], rely on fixed relation types and as such are also undesirable for the contest task.

We ultimately found that the best performance was achieved by maintaining a high level of simplicity and utilising a pipeline-based approach. Our system is built using well-established natural language processing frameworks such as NLTK[3] and SpaCy[4], and makes use of standard techniques such as tokenisation, part-of-speech (POS) tagging, named entity recognition, coreference resolution, and noun/verb phrase chunking. We incorporate several of our own algorithms in order to address the aforementioned shortcomings of NER on domain-specific data.

B. Triple extraction system

Our triple extraction system adopts a pipeline-based approach in order to convert a document into a set of triples. It comprises seven distinct stages, as shown in Figure 1.

Text cleaning: Text data is cleaned to manage special characters such as hyphen and quotation marks and also break sentences joined together with no space between them.

Text processing: The text is processed through tokenisation, POS tagging, entity recognition and dependency parsing steps using SpaCy. The results are shown in Table I for the following text: *Ford Motor Company is an American*

[1]https://www.clips.uantwerpen.be/conll2003/ner/
[2]https://catalog.ldc.upenn.edu/LDC2013T19
[3]https://www.nltk.org/
[4]https://spacy.io/

2374-8486/19/$31.00 ©2019 IEEE
DOI 10.1109/ICDM.2019.00205

multinational automaker that has its main headquarters in Dearborn, Michigan, a suburb of Detroit. The company was founded by Henry Ford and incorporated on June 16, 1903.

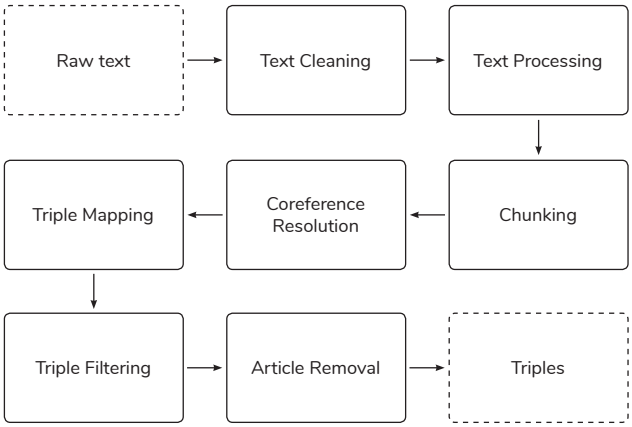

Fig. 1. A diagram of the core components of our triple extraction system.

Chunking: Noun phrases (NPs) and verb phrases are chunked, as shown in Table II. Noun chunks are phrases that have a noun and the words describing the noun. For example, *an American multinational automaker* and *a suburb of Detroit*. We also implemented the chunking of action words, so that verb phrases can contain verbs, particles and/or adverbs that represent more meaningful relations between entities. For example, *was founded by* and *incorporated on*.

Algorithm 1 Chunking of noun phrases and verb phrases

procedure CHUNKPHRASES(*document*)
  for each *sentence in document* **do**
  ▷ Chunk noun phrases (NPs) and tag as *ENTITY*
    chunk *NPs* ▷ NP
    chunk $'('+NP+')'$ ▷ (NP)
    chunk $NP+' of'+NP$ ▷ NP of NP
    chunk $NP+NP$ ▷ NP NP
  ▷ Chunk verb phrases and tag as *VERB*
    chunk $VERB+PART$ ▷ verb + particle
    chunk $VERB+ADP$ ▷ verb + adpositions
    chunk $ADP+VERB$ ▷ adpositions + verb
    chunk $PART+VERB$ ▷ particle + verb
    chunk $VERB+VERB$ ▷ verb + verb
  return *document* ▷ Document with phrase chunks

Coreference Resolution: A list of coreferenced items is created using NeuralCoref[5]. For our example the following two coreference items are identified: *Ford Motor Company - its* and *Ford Motor Company - The company*. Coreference items are resolved on the triples by replacing the original phrase with the referred phrase for each item. For example, *The company* will be replaced by *Ford Motor Company*. In the case of pronouns such as *its, her, his* or *their*, we ignore the coreference items. As we prefer *main headquarters* over *Ford Motor Company main headquarters*, since *main headquarters* will be connected to *Ford Motor Company* by the triples.

Triple Mapping: Triples are created from the sentences in *head, relation, tail* format using Algorithm 2. First, head and tail entities are extracted with their relations from the sentences and creates a list of triples. Second, a graph is created from those triples to uncover the relations among named entities in separate sentences. Based on the relations of prepositions such as *in, on, at*, more triples are created to provide more links between named entities in the graph. Finally, the triples created by these two steps are joined to make the full list of triples for the given text.

Triple Filtering: To improve the quality of the triples, the filtering is performed to remove any triple with a stop word as a head entity. The stop words include NLTK stop words, names of days (Monday to Sunday) and names of months (January to December).

Article Removal: To clean the entities we removed some tokens including articles (e.g., a, an, the), possessive pronouns (e.g., its, their) and demonstrative pronouns (e.g., that, these) from the head and tails of each triple.

C. Visualisation system

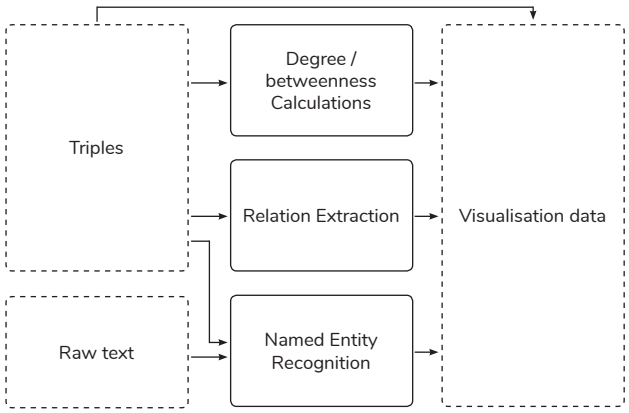

Fig. 2. The additional stages performed by our system prior to visualisation in order to display more detailed information about each triple.

Our visualisation system, which displays the results of the triple extraction system, performs three additional techniques in order to maximise the information displayed via our web application. After the triples have been generated as per Section I-B, they are post-processed and appended with the degree/betweenness of the head and tail nodes, structured relation(s) corresponding to the verb relation of each triple, and the named entity classes of each head and tail. This process is displayed in Figure 2. The source code of our visualisation system is available on Github[6].

The **degree/betweenness calculation** determines the degree and betweenness centrality of the head and tail of each

[5]https://github.com/huggingface/neuralcoref

[6]https://github.com/Michael-Stewart-Webdev/text2kg-visualisation

Token Id	Token	Entity Type	IOB	Coarse Grained POS	POS	Start	End	Dependency
0	Ford	ORG	B	PROPN	NNP	0	3	compound
1	Motor	ORG	I	PROPN	NNP	5	9	compound
2	Company	ORG	I	PROPN	NNP	11	17	nsubj
3	is		O	VERB	VBZ	19	20	ROOT
4	an		O	DET	DT	22	23	det
5	American	NORP	B	ADJ	JJ	25	32	amod
6	multinational		O	ADJ	JJ	34	46	amod
7	automaker		O	NOUN	NN	48	56	attr
8	that		O	DET	WDT	58	61	nsubj
9	has		O	VERB	VBZ	63	65	relcl
10	its		O	DET	PRP	67	69	poss
11	main		O	ADJ	JJ	71	74	amod
12	headquarters		O	NOUN	NN	76	87	dobj
13	in		O	ADP	IN	89	90	prep
14	Dearborn	GPE	B	PROPN	NNP	92	99	pobj
15	,		O	PUNCT	,	100	100	punct
16	Michigan	GPE	B	PROPN	NNP	102	109	appos
17	,		O	PUNCT	,	110	110	punct
18	a		O	DET	DT	112	112	det
19	suburb		O	NOUN	NN	114	119	dobj
20	of		O	ADP	IN	121	122	prep
21	Detroit	GPE	B	PROPN	NNP	124	130	pobj
22	.		O	PUNCT	.	131	131	punct
23	The		O	DET	DT	133	135	det
24	company		O	NOUN	NN	137	143	nsubjpass
25	was		O	VERB	VBD	145	147	auxpass
26	founded		O	VERB	VBN	149	155	ROOT
27	by		O	ADP	IN	157	158	agent
28	Henry	PERSON	B	PROPN	NNP	160	164	compound
29	Ford	PERSON	I	PROPN	NNP	166	169	pobj
30	and		O	CCONJ	CC	171	173	cc
31	incorporated		O	VERB	VBD	175	186	conj
32	on		O	ADP	IN	188	189	prep
33	June	DATE	B	PROPN	NNP	191	194	pobj
34	16	DATE	I	NUM	CD	196	197	nummod
35	,	DATE	I	PUNCT	,	198	198	punct
36	1903	DATE	I	NUM	CD	200	203	nummod
37	.		O	PUNCT	.	204	204	punct

TABLE I

TEXT PROCESSING: TOKENISATION, POS TAGGING, ENTITY RECOGNITION, AND DEPENDENCY PARSING.

Sent #	Phrase #	Phrase	Type
0	0	Ford Motor Company	ENTITY
0	1	is	VERB
0	2	an American multinational automaker	ENTITY
0	3	that	DET
0	4	has	VERB
0	5	its main headquarters	ENTITY
0	6	in	ADP
0	7	Dearborn	ENTITY
0	8	,	PUNCT
0	9	Michigan	ENTITY
0	10	,	PUNCT
0	11	a suburb of Detroit	ENTITY
0	12	.	PUNCT
1	13	The company	ENTITY
1	14	was founded by	VERB
1	15	Henry Ford	ENTITY
1	16	and	CCONJ
1	17	incorporated on	VERB
1	18	June 16, 1903	ENTITY
1	19	.	PUNCT

TABLE II

CHUNKS OF NOUN PHRASES AND VERB PHRASES.

triple. In graph theory, *degree* refers to the number of edges connected to a node [11]. For triples, this directly corresponds to the number of triples in which each phrase appears. *Betweenness centrality*, on the other hand, measures the extent to which each vertex lies along the paths between other vertices. Phrases that exert a high degree of influence over the flow of the graph, such as company names ("Ford", "BYD") tend to have a high betweenness value and are hence more important than other terms. Incorporating the degree and betweenness calculations allows for this information to be conveyed in the visualisation.

The **relation extraction** component maps the relation phrase of each triple to one or more structured relation types. This allows for the graph visualisation to display structured relation types when desired by the user. Our system currently maps each relation phrase to its corresponding SemEval [12] relation. To accomplish this we use an attention-based bidirectional Long Short-Term Memory (LSTM) model [13], which maps a sequence of words padded with entity markers ($\langle e_1 \rangle$ and $\langle e_2 \rangle$) to a fixed relation type. We create sequences using the head and tail of each triple as $\langle e_1 \rangle$ and $\langle e_2 \rangle$ respectively, and feed them into a pretrained model (trained on the SemEval 2010 Task 8 dataset) to obtain the corresponding SemEval relation. SemEval contains nine types of semantic relations and an additional type for other relations.

Finally, the **named entity recognition** (NER) component

Algorithm 2 Triple mapping algorithm

procedure GETTRIPLES(*document*)
  for each *sentence in document* **do**
    $relations \leftarrow verbs + prepositions + postpositions$ ▷ Select relations such as *showcased, has, in, to, during*
    for each r *in relations* **do**
      $heads \leftarrow$ *entities on the left side of* r ▷ Get the head entities for the relation *r*
      $tails \leftarrow$ *entities on the right side of* r ▷ Get the tail entities for the relation *r*
      for each h *in heads* **do**
        for each t *in tails* **do**
          $triples \leftarrow triples + [h, r, t]$ ▷ Add *[head, relation, tail]* to the list of triples
  return *triples* ▷ Return the list of triples
procedure EXTRACTTRIPLES(*document*)
  ▷ Extract triples from the document at the sentence level
  $triples \leftarrow$ GETTRIPLES(*document*)
  ▷ Extract the triples at the document level using the graph shortest paths
  $G \leftarrow$ *create graph(triples)* ▷ Build a graph from the triples using NetworkX package
  $paths \leftarrow$ *get shortest paths(G)* ▷ Get all shortest paths between named entities
  for each h, t *in pairs of named entities* **do**
    if *h and t connected by a path using 'in', 'at', 'on' prepositions* **then**
      $triples \leftarrow triples + [h,' in', t]$ ▷ Add *[head, 'in', tail]* to the list of triples
  return *triples* ▷ Return the full list of triples

determines the semantic type of the head and tail of each triple. We label each phrase with one of five types: `PER`, `ORG`, `LOC`, `MISC`, and `O`, based upon the Wikipedia NER scheme [14]. The raw text is first labelled via SpaCy, yielding a set of entities E. Each phrase (head and tail) in each triple are then compared to every entity $e \in E$ and assigned the same label as e when the phrase is highly similar to e in terms of edit distance. One caveat of performing the NER after the triple extraction pipeline is that there is no contextual information passed to the named entity recognition model. However, applying NER immediately prior to visualisation allows for a greater level of abstraction and flexibility.

II. Evaluation and Conclusion

A. Triple Extraction

In order to evaluate the quality of our triple extraction system, we consider the following two sentences: *Ford Motor Company is an American multinational automaker that has its main headquarters in Dearborn, Michigan, a suburb of Detroit. It was founded by Henry Ford and incorporated on June 16, 1903.*

Table III displays the *subject, predicate, object* triples from our triple extraction system and shows the additional information provided by the visualisation system: the SemEval relation type, the named entity types of the heads and tails, and the degree and betweenness of each head and tail.

The triples show some of the notable strengths of our model: the chunking component ensures useful phrases such as "Ford Motor Company" and "Henry Ford" appear in multiple triples. Furthermore, our system is able to extract useful triples with "in" relations via the triple mapping component, such as `(Ford Motor Company, in, Dearborn)`.

B. Coreference Resolution

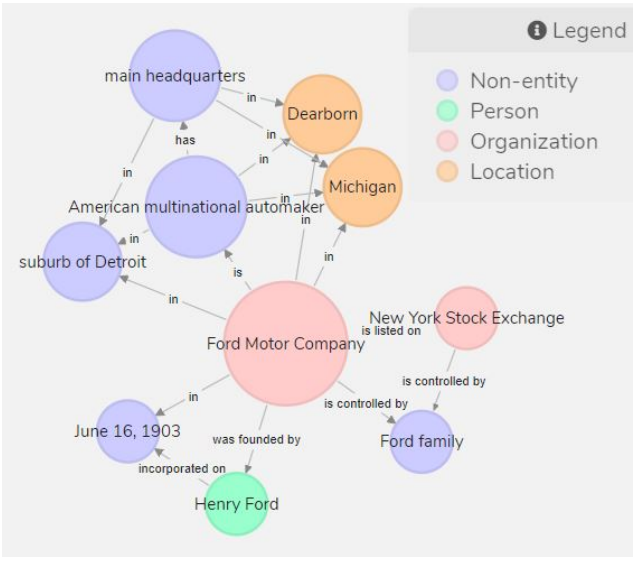

Fig. 3. An example graph generated by our triple extraction system. The nodes are coloured based on their named entity types. The node sizes are based on their degree centrality values.

To highlight the effectiveness of our coreference resolution component, we introduce an additional sentence to our example so that it becomes:
Ford Motor Company is an American multinational automaker that has its main headquarters in Dearborn, Michigan, a suburb of Detroit. It was founded by Henry Ford and incorporated on June 16, 1903. The company is listed on the

New York Stock Exchange and it is controlled by the Ford family.

Figure 3 shows the result of visualising the above sentences via our web application. The underlined words (*Ford Motor Company* in sentence 1, *It* in sentence 2 and *The company* and *it* in sentence 3) represent the same entity *Ford Motor Company*. The visualisation in Figure 3 clearly shows *Ford Motor Company* as the shared entity between the three sentences. The node *Ford Motor Company* appears the biggest among all nodes in the graph, to represent the highest degree centrality of the node in that graph.

C. Conclusion

In conclusion, our system uses a pipeline-based approach to extract a set of triples from a given document. It offers a simple and effective solution to the challenge of knowledge graph construction from domain-specific text. It also provides the facility to visualise useful information about each triple such as the degree, betweenness, structured relation type(s), and named entity types.

It is important to note that the graph edit distance metric that is commonly used to automatically evaluate the quality of triples is only capable of structural analysis. In order to improve the metric it could be combined with meaningful semantic measures such as those present in the machine translation and image captioning domains (e.g. SPICE [15]). Another option would be to incorporate a simple sum of word embeddings over each triple so that semantic information is captured by the metric.

In future we plan to continue working on our end-to-end deep learning-based triple extraction model.

III. External Resources

Our triple extraction system uses the aforementioned NLTK [3] and SpaCy [4] at various stages throughout the pipeline.

Our visualisation system is written in Flask[7]. The front-end visualisations are written primarily in D3.js[8]. The attention-based Bi-LSTM [13] for relation extraction is implemented in Tensorflow [16], and trained on the SemEval 2010 Task 8 dataset [12]. The degree and betweenness calculations are performed via NetworkX[9].

Acknowledgement

We would like to thank our team members Morgan Lewis and Thomas Smoker, who are in the early stage of their PhD candidatures, for their contributions on literature search.

Triple			Additional information					
Head ($_H$)	Relation ($_R$)	Tail ($_T$)	SemEval Relation	Type$_H$	Type$_T$	Deg$_H$	Deg$_T$	Betw$_H$	Betw$_T$
Ford Motor Company	in	Dearborn	Content-Container	ORG	LOC	6	3	11.0	0.75
Ford Motor Company	in	Michigan	Content-Container	ORG	LOC	6	3	11.0	0.75
Ford Motor Company	in	suburb of Detroit	Member-Collection	ORG	O	6	3	11.0	0.75
Ford Motor Company	in	June 16, 1903	Component-Whole	ORG	O	6	2	11.0	0.0
Ford Motor Company	is	American multinational automaker	Instrument-Agency	ORG	O	6	5	11.0	1.75
Ford Motor Company	was founded by	Henry Ford	Product-Producer	ORG	PER	6	2	11.0	0.0
American multinational automaker	in	Dearborn	Member-Collection	O	LOC	5	3	1.75	0.75
American multinational automaker	in	Michigan	Member-Collection	O	LOC	5	3	1.75	0.75
American multinational automaker	in	suburb of Detroit	Member-Collection	O	O	5	3	1.75	0.75
American multinational automaker	has	main headquarters	Cause-Effect	O	O	5	4	1.75	1.0
Henry Ford	incorporated on	June 16, 1903	Component-Whole	PER	O	2	2	0.0	0.0
main headquarters	in	Dearborn	Content-Container	O	LOC	4	3	1.0	0.75
main headquarters	in	Michigan	Content-Container	O	LOC	4	3	1.0	0.75
main headquarters	in	suburb of Detroit	Member-Collection	O	O	4	3	1.0	0.75

TABLE III
EXAMPLE TRIPLES PRODUCED BY OUR TRIPLE EXTRACTION SYSTEM, ALONG WITH THE ADDITIONAL INFORMATION APPENDED TO EACH TRIPLE VIA OUR VISUALISATION SYSTEM.

[7]https://www.fullstackpython.com/flask.html
[8]https://d3js.org/
[9]https://networkx.github.io

REFERENCES

[1] E. F. T. K. Sang and F. De Meulder, "Introduction to the conll-2003 shared task: Language-independent named entity recognition," *CoNLL-2003*, 2003.

[2] R. Weischedel, M. Palmer, M. Marcus, E. Hovy, S. Pradhan, L. Ramshaw, N. Xue, A. Taylor, J. Kaufman, M. Franchini *et al.*, "Ontonotes release 5.0 ldc2013t19," *Linguistic Data Consortium, Philadelphia, PA*, 2013.

[3] S. Bird, E. Klein, and E. Loper, *Natural language processing with Python: analyzing text with the natural language toolkit*. " O'Reilly Media, Inc.", 2009.

[4] M. Honnibal, "Spacy," 2017. [Online]. Available: https://explosion.ai/blog/introducing-spacy

[5] J. R. Finkel, T. Grenager, and C. Manning, "Incorporating non-local information into information extraction systems by gibbs sampling," in *Proceedings of the 43rd annual meeting on association for computational linguistics*. Association for Computational Linguistics, 2005, pp. 363–370.

[6] M. Gardner, J. Grus, M. Neumann, O. Tafjord, P. Dasigi, N. F. Liu, M. Peters, M. Schmitz, and L. S. Zettlemoyer, "Allennlp: A deep semantic natural language processing platform," 2017.

[7] M. E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer, "Deep contextualized word representations," *arXiv preprint arXiv:1802.05365*, 2018.

[8] C. Niklaus, M. Cetto, A. Freitas, and S. Handschuh, "A survey on open information extraction," *arXiv preprint arXiv:1806.05599*, 2018.

[9] L. Cui, F. Wei, and M. Zhou, "Neural open information extraction," *arXiv preprint arXiv:1805.04270*, 2018.

[10] N. Kertkeidkachorn and R. Ichise, "T2kg: An end-to-end system for creating knowledge graph from unstructured text," in *Workshops at the Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

[11] R. Diestel, "Graph theory. 2005," *Grad. Texts in Math*, vol. 101, 2005.

[12] I. Hendrickx, S. N. Kim, Z. Kozareva, P. Nakov, D. Ó Séaghdha, S. Padó, M. Pennacchiotti, L. Romano, and S. Szpakowicz, "Semeval-2010 task 8: Multi-way classification of semantic relations between pairs of nominals," in *Proceedings of the Workshop on Semantic Evaluations: Recent Achievements and Future Directions*. Association for Computational Linguistics, 2009, pp. 94–99.

[13] P. Zhou, W. Shi, J. Tian, Z. Qi, B. Li, H. Hao, and B. Xu, "Attention-based bidirectional long short-term memory networks for relation classification," in *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, 2016, pp. 207–212.

[14] J. Nothman, N. Ringland, W. Radford, T. Murphy, and J. R. Curran, "Learning multilingual named entity recognition from wikipedia," *Artificial Intelligence*, vol. 194, pp. 151–175, 2013.

[15] P. Anderson, B. Fernando, M. Johnson, and S. Gould, "Spice: Semantic propositional image caption evaluation," in *European Conference on Computer Vision*. Springer, 2016, pp. 382–398.

[16] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard *et al.*, "Tensorflow: A system for large-scale machine learning," in *12th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 16)*, 2016, pp. 265–283.