


BLCU_NLP at SemEval-2019 Task 7: An Inference Chain-based GPT Model for Rumour Evaluation

Ruoyao Yang, Wanying Xie, Chunhua Liu, Dong Yu 

Beijing Language and Culture University, Beijing, China

{yangruoyao97, xiewanying07, chunhualiu596}@gmail.com
yudong@blcu.edu.cn

Abstract

Researchers have been paying increasing attention to rumour evaluation due to the rapid spread of unsubstantiated rumours on social media platforms, including SemEval 2019 task 7. However, labelled data for learning rumour veracity is scarce, and labels in rumour stance data are highly disproportionate, making it challenging for a model to perform supervised-learning adequately. We propose an inference chain-based system, which fully utilizes conversation structure-based knowledge in the limited data and expand the training data in minority categories to alleviate class imbalance. Our approach obtains 12.6% improvement upon the baseline system for subtask A, ranks 1st among 21 systems in subtask A, and ranks 4th among 12 systems in subtask B.

1 Introduction

With the universality of the Internet, social media has become the main channel for acquiring and exchanging information. However, the free flow of information has given rise to the prevalence of rumours, among which fake ones are harmful since they are generally convincing and hard to distinguish. To address this problem, we need automatic rumour veracity classification on social media. A large amount of rumour stance instances on social media have been employed to assist the model in making better predictions regards the rumour's veracity.

Rumour stance classification and rumour veracity classification are two subtasks of SemEval 2017 Task 8 (Derczynski et al., 2017) and SemEval 2019 Task 7 (Gorrell et al., 2018). Subtask A predicts the stance of a post replying to a rumourous post, in terms of supporting, denying, querying and commenting the rumour. Subtask B anticipates the veracity of a rumour as true or false

given the rumourous post and a set of additional resources.

Apart from variations in models, research in this area mainly focuses on the special characteristics of data coming from social media: conversation structure, rich intrinsic features, skewed distribution toward the *comment* class in rumour stance data and scarcity of available data for rumour veracity classification. While most pioneering works treated rumour evaluation as a single-tweet task, attempts to utilize the conversation structure included pairing source and replies together to make up input (Singh et al., 2017), and adopting the full conversation thread as input in the time sequence (Kochkina et al., 2017). With the realization of rich features hidden in tweet contexts, Qazvinian et al. (2011) was one of the first who extracted them and combined them with model input. The feature sets were augmented during the following work. In trying to acquire more comprehensive information, not only features of the tweet for prediction were taken into consideration, but also features from its conversation context (Enayet and El-Beltagy, 2017). To address the class imbalance problem, Wang et al. (2017) transformed subtask A into a two-step classification task: they first classified comments and non-comments, and then categorized non-comments into the other three classes. Finally, in order to make up for the absence of abundant accessible training data in the rumour veracity classification task, external resources usage such as Wikipedia dumps and news articles was encouraged in both RumourEval contests.

In our work, we find that simply taking the whole conversation as input is inadequate. We have to recognize the role each part assumes in the conversation thread and mark them accordingly in the input. Instead of filtering features solely based on the system performance, we pre-

fer to run a feature selection before adding to the system and choose those that can bring a high deviation degree between data categories. Following the feature extraction work of Enayet and El-Beltagy (2017), we consider introducing more features from the conversation context to further assist model judgment. We alleviate class imbalance in stance classification by expanding training data in the under-represented classes with pre-screened external data from similar datasets. At last, we approach the data insufficiency issue by setting an average length limit and cutting the overlength ones to enlarge training data.

We propose an inference chain-based system for this paper. A conversation thread starts with a source tweet, and follows by replies, in which each one responds to an earlier one in time sequence. When we infer the stance of one tweet, the source or earlier replies in the same thread can give abundant additional hints. Therefore, we take each conversation thread as an inference chain and concentrate on utilizing it to solve the data issues discussed earlier.

Our approach for both tasks is fine-tuned on Generative Pre-trained Transformer (OpenAI GPT) (Radford et al., 2018), a model that has performed well in 9 NLP tasks. Our work primarily focuses on subtask A rumour stance classification, in which we expand training data from similar datasets, extract features and join separate parts to form input according to their roles in inference chain. For subtask B rumour veracity classification, we apply similar feature extraction and input concatenation process, except for replacing the data expansion step with data slicing. With the above implementation, our model outperforms all other systems in subtask A and places 4th in subtask B in SemEval 2019.

2 System Description

We propose a system that focuses on inference chain-based knowledge enhancement. The operations involved are displayed in Figure 1. We first perform data preprocessing on the raw dataset (Section 2.1) to fetch tweet content and facilitate the subsequent feature extraction step. Then we implement two data extension mechanisms on the training data: to relieve class imbalance toward one category in subtask A, we expand training data with external datasets (Section 2.2); to alleviate data sparsity and try to avoid under-fitting

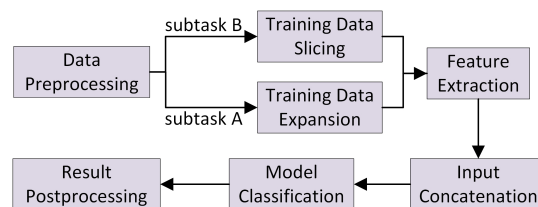


Figure 1: The operational flowchart for our system.

on the training set in subtask B, we set a length limit for each part of the input and split data instances accordingly (Section 2.3) to enlarge the amount of data. In trying to better utilize inference chain-based features, we extract word-level and tweet-level features and filter them for each subtask (Section 2.4). Finally, we concatenate each part in the inference chain with its features together (Section 2.5) and feed them in the model. After model classification, we adjust some of the results (Section 2.6) according to the organizers’ requirements.

An illustration of our system is shown in Figure 2. How our base model GPT performs on the two classification tasks is depicted in the upper left corner. The right side presents how we organize our input for subtask A and B. In the lower left corner, we give an example of an inference chain (conversation thread) in the training data. To help understand, we define each part of it for the rest of this passage. For subtask A, we divide an inference chain into four parts: *source tweet*, *other tweets*, *parent tweet* and *target tweet*. For subtask B, an inference chain constitutes of a *source tweet* and a *thread content*. To better clarify, a *thread content* is defined as a whole inference chain excluding the source tweet, and we also define a whole inference chain excluding the target tweet as a *conversation context*. As depicted in the upper left corner, the goal is to predict the stance of a target tweet toward a rumour for subtask A, and the veracity of a rumour (usually contained in the source tweet) for subtask B.

2.1 Data Preprocessing

After extracting the tweet content out of the original data, we first perform word tokenization with Stanford CoreNLP¹, Spacy² and NLTK tools³, among which the result of Stanford CoreNLP

¹<https://stanfordnlp.github.io/CoreNLP/>

²<https://spacy.io/>

³<http://www.nltk.org/>

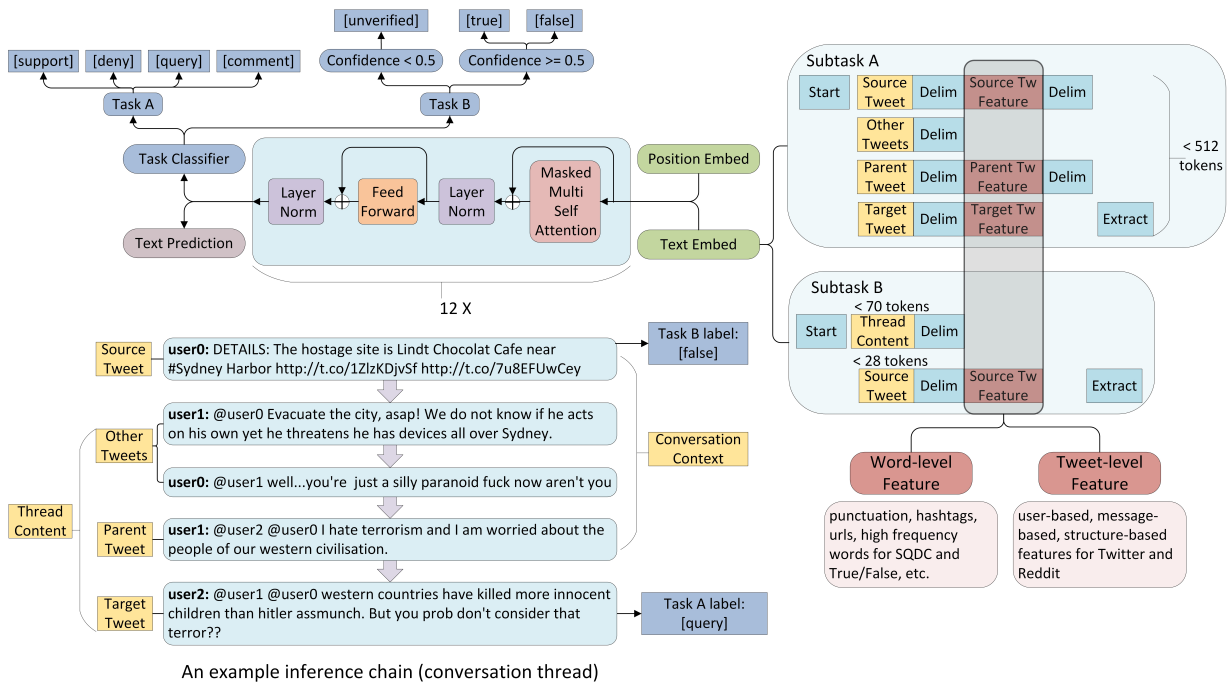


Figure 2: An overview of our system. The upper left corner is a diagram of how our base model GPT performs on the two classification tasks. The upper right corner is how we organize our input for subtask A and B. The lower left corner is a data example which helps explain “inference chain”, “source tweet”, “other tweets”, “parent tweet”, “target tweet”, “thread content” and “conversation context” in the input. The lower right corner is the features we combined in the input.

proves to be the best. Then we fix some tokenization inaccuracies in the result. Blanks, emoticons and uncommon punctuations are removed to reduce the amount of Out of Vocabulary (OOV). Besides, we transform all user mentions into “@” and all URLs into “urlurlurl” in order to increase the repetition rate of these features and allow the model to learn them better. Finally, we convert all letters into their lower case for the same purpose.

2.2 Training Data Expansion

We find that the data distribution in the training set is skewed toward *comment* in subtask A, which explains why it is hard for the model to reach high precision and recall scores in the other three classes. Thus we expand the training data with datasets on similar tasks with labels corresponding to the three minority classes in seeking for better class balance and more sufficient training.

For *support* and *deny*, we take each claim as both the target tweet and its conversation context in stance classification datasets SemEval 2016 task 6 dataset (Mohammad et al., 2016), Emergent (Ferreira and Vlachos, 2016), and twitter sentiment analysis dataset sentiment140 (Go et al., 2009). For *query*, we use passages as the conversa-

Training Set			
Class	Origin	Extended	Sum
support	925(18%)	797	1722(23%)
deny	378(7%)	696	1074(14%)
query	395(8%)	912	1307(17%)
comment	3519(67%)	0	3519(46%)

Table 1: Distribution of tweets between classes before and after data expansion in the training set for subtask A.

tion context, unanswerable questions as the target tweet in reading comprehension datasets SQuAD 2.0 (Rajpurkar et al., 2018) and CoQA (Reddy et al., 2018).

The acquirement of extended data takes two steps. We first calculate the sentence similarity of each data piece in external datasets with all sentences in the original dataset in terms of Levenshtein Distance, and only keep instances whose minimal distances with the original dataset are below 0.7. Then we test them in a model trained with the original dataset. If the model prediction is identical to the label we expect, we append this instance in our training set for subtask A. The data distribution before and after data expansion is

shown in Table 1.

2.3 Training Data Slicing

The training set for subtask B only has 327 pieces. Since the organizers define subtask B as a binary classification task, which classifies instances into two categories: *true* and *false*, and takes pieces whose confidence scores below 0.5 as *unverified*, the *unverified* class in the training set is entirely useless, which takes up 98 pieces. Discard of this class only makes the data scarcity problem worse and may lead to under-fitting on the training set.

We try to extend the training set from a different angle. We look at the distribution of sequence length and set a length restriction for each conversation context to 70 tokens and each target tweet to 28 tokens. For those that exceed the limit in the training set, we truncate each piece to the edge to create multiple instances and thus enlarge the training set. For the development set and test set, the length restrictions are also set, but only the first truncated piece for each instance is taken as input. Although data slicing may hurt long-distance dependency, the experiment result shows that this method performs better than the original.

2.4 Feature Extraction

Inspired by the features extracted by Kochkina et al. (2017) and Bahuleyan and Vechtomova (2017), we collect 56 word-level features and 16 tweet-level features. For word-level features, we calculate their distribution percentage on the four categories in subtask A and the three categories in subtask B in the training and development set, and only apply those that mark a clear distinction between the classes for each subtask. For the numerical tweet-level features, we cluster each one into several groups according to their values and determine a common value for the whole group. Where to add the features is illustrated on the right side in Figure 2.

After selection, the word-level features we apply to subtask A and B are as below:

Subtask A: Whether the tweet content has question marks, hashtags, URLs, “RT”(refers to retweet), positive words, negative words, swear words, query words, forbidding verbs, accusing verbs, complaining verbs, warning verbs, permitting verbs, praising verbs, etc.

Subtask B: Whether the tweet content has exclamation marks, positive words, negative words, query words, false synonyms, false antonyms,

declaring verbs, confirming verbs, arguing verbs, etc.

The tweet-level features we add for subtask A are as below:

Features for both Twitter and Reddit: Tweet favourite count, tweet depth in the thread and whether the user has user description.

Twitter-specific features: User-related features include whether the user is verified, user-related URLs, whether the user uses the default profile, user followers count, user friends count, user listed groups count, and user statuses count. Tweet-related features include tweet retweet count.

Reddit-specific features: Whether the tweet has self-text, tweet kind, whether the tweet is archived, whether the tweet is edited, whether the user is a submitter.

We’ve tried to add the above tweet-level features to input for subtask B, but unfortunately observe no performance improvement in our experiment, so the official submission for subtask B involves no tweet-level features.

As indicated in Enayet and El-Beltagy (2017), whether the users of the source and parent tweet are verified are useful in performance improvement for subtask A. So we speculate that word-level features and other tweet-level features may also be necessary hints for model prediction. We find that the model always creates mispredictions because the stance of the target tweet toward the rumour is indirect. Since many tweets express direct stances toward their parent tweets, we can infer their stances toward the rumour with the help of their parent. In addition, the model often mistakes the stance of target tweet toward source tweet as its stance toward the rumour. But in cases where the source tweet expresses *deny* towards the rumour, the two stances above are not consistent. Therefore, parent tweet and source tweet assume significant roles in an inference chain and we need to acquire more knowledge from them to improve the prediction accuracy of the target tweet. So we apply all filtered word-level and tweet-level features for the source, the parent and the target to the input for subtask A.

2.5 Input Concatenation

As described above, the model is likely to make correct predictions when it learns information from earlier tweets in the inference chain. We plan to concatenate the contexts in the inference chain

with the target tweet to make up for input. The input concatenation method is depicted in the upper right corner of Figure 2. We concatenate *source tweet* with *thread content* to form an input structure for subtask B. We apply two methods in subtask A. At first we concatenate the *conversation context* and the *target tweet* with the same structure as subtask B. However, this method often results in the model’s confusion between the stance of the target tweet toward the source and the parent. Thus, we decide to mark the position of the source and the parent in the conversation thread with delimiters and connect them with the rest of the conversation (other tweets) together. Each part is marked by delimiters and features are inserted behind them in the input.

2.6 Model

We employ OpenAI GPT that has been pretrained on BooksCorpus (Zhu et al., 2015) as our base model for task-specific fine-tuning. We’ve tried different activation functions, optimizers and hyperparameters but observe no performance improvement. So we use the default model configuration for the official submission. GPT requires its input length to be less than 512 tokens. For inputs that exceed this limit, we choose to cut off the “other tweets” sequence to fit this restriction.

Result Postprocessing As required by the organizers, we transform the labels of all blank instances into *comment* for subtask A, and take pieces whose confidence levels are below 0.5 as *unverified* in subtask B in the model prediction results.

3 Evaluation

We conduct experiments on data expansion, input format adjustment and word & tweet-level feature adding for subtask A, and perform data slicing and feature adding for subtask B. The dataset we used for this task is obtained from Zubiaga et al. (2016). We primarily focus on achieving a higher macro f1 score for both subtasks. We also display the comparison of the results on the test set between our official system and the baseline systems provided by the organizer.

3.1 Experiment Results

The following are the steps we conduct experiments toward creating the system for official submission.

Subtask A Our experiments conduct on the development set for subtask A is shown in Table 2. With the first input format (A), we achieve an initial result of 53.48%. Combining word-level, tweet-level features (B), and processing data expansion (C) brings an increase of 0.99% and 1.61% respectively. After converting to the second input format (D), we’ve seen a rise of 2.03% compared to A. Training data expansion (E) is also implemented on this input, but a decrease of 0.86% is observed. We suspect that after enhancing the percentage of the source and parent tweet in new input, data dissimilarity brought by the external dataset is aggravated. However, this inferior position is reversed by employing features. Word-level features (F) alone bring a 2.04% growth. Though tweet-level features alone haven’t led in any extra increase, adding them together with word-level features (G) produces a result above 56%. Our final result for subtask A is ensembled on three runs (F1, F2, G) that achieve the best performance on the development set.

Subtask A	
Systems	MacroF
A. GPT(1st input)	0.5348
B. GPT(1st input)+WF+TF	0.5447
C. GPT(1st input)+DE	0.5509
D. GPT(2nd input)	0.5551
E. GPT(2nd input)+DE	0.5465
F1. GPT(2nd input)+DE+WF	0.5644
F2. (another run)	0.5669
G. GPT(2nd input)+DE+WF+TF	0.5631

Table 2: Ablation results on the development set for subtask A. MacroF means macro f1 score. 1st input consists of conversation context and target tweet, while a 2nd input constitutes of source tweet, other tweets, parent tweet and target tweet. WF and TF refer to word-level and tweet-level features respectively. DE represents data expansion.

Subtask B Our experiments conducted on the development set for subtask B is shown in Table 3. Word features bring an increase of 3.71%. Tweet features prove to be disruptive and lead to a drop of up to 11.6%. So we discard this type of feature in subtask B entirely. The result we’ve submitted for subtask B is ensembled on two runs (B) that get the highest values on the development set.

Final Result Our final result on the test set and the comparison with the baseline systems are shown in Table 4. BranchLSTM and NileTMRG are two baselines implemented by the organizers.

Subtask B	
Systems	MacroF
A. GPT+DS	0.4701
B. GPT+DS+WF	0.5072
C. GPT+DS+WF+TF	0.4143

Table 3: Ablation results on the development set for subtask B. DS refers to training data slicing. WF and TF are the same meaning as Table 2.

Our system is 12.6% higher than the baseline system in macro f1 for subtask A, but 8.3% and 5.6% lower in macro f1 for subtask B compared with BranchLSTM and NileTMRG respectively. Our system ranks first in subtask A and fourth in subtask B.

Official Submission			
	Subtask A	Subtask B	
System	MacroF	MacroF	RMSE
BranchLSTM	0.493	0.336	0.781
NileTMRG		0.309	0.769
Our System	0.6187	0.2525	0.8179

Table 4: Official submission results on the test set for our system and the organizers’ baselines.

3.2 Error Analysis

We perform error analysis for the results on the test set. Classification reports for both subtasks are provided in Table 5. The problem for subtask A lies in that the model often confused the *comment* class with the other three classes. The possible reason can be the relatively larger proportion of *comment* data. None of the classes performs well in prediction for subtask B, though results are better for the *true* class comparing with the other two classes. Precision, recall and macro f1 scores go extremely low for the *unverified* class. A reasonable explanation may be the *unverified* class is not directly acquired from the model but comes from the other two classes.

3.3 Comparison with ESIM

Since rumour evaluation can be seen as a task that given a target tweet and its background conversation, infer the target label of this tweet, they can be treated as a subtask of natural language inference, so we employ ESIM (Chen et al., 2016) in this task.

The results are shown in Table 6. The input format is the same as subtask B in GPT. We also try to apply features and change word embeddings in ESIM. But no results reach the result in GPT with

Subtask A				
	prec.	rec.	f1	distribution
support	0.89	0.93	0.91	1476
deny	0.45	0.51	0.48	101
query	0.62	0.59	0.60	93
comment	0.66	0.38	0.48	157
Subtask B				
	prec.	rec.	f1	distribution
true	0.46	0.47	0.47	40
false	0.22	0.13	0.16	31
unverified	0.09	0.20	0.13	10

Table 5: Classification report on the test set for both subtasks.

an equivalent configuration. So the base model we employ for the system is GPT instead of ESIM.

Subtask A	
Systems	MacroF
A. ESIM(glove)	0.434
B. ESIM(glove)+WF+TF	0.452
C. ESIM(google news)+WF+TF	0.466
Subtask B	
Systems	MacroF
A. ESIM(glove)	0.473

Table 6: ESIM results on the development set for both subtasks.

4 Conclusions

We introduce a framework with a strong focus on inference chain-based knowledge enhancement for determining rumour stance and veracity in SemEval 2019 task 7. In order to address the problems of class imbalance, training data scarcity, model’s insufficient learning of features and tree-structured conversations, we employ data expansion, data slicing, feature extraction, and input concatenation mechanisms in our system respectively. Our system takes first place in subtask A and fourth place in subtask B.

In future, we would like to introduce synonyms, tweet similarity and sentiment features in our model to further facilitate the recognition of relations between tweets. We will also utilize the prediction results from stance classification, expand training data with external datasets and introduce additional knowledge base such as Wikipedia to assist the model prediction in rumour veracity classification.

Acknowledgments

This work is funded by Beijing Advanced Innovation for Language Resources of BLCU (TYR17001J); the Fundamental Research Funds for the Central Universities in BLCU (No.17PT05) and the BLCU Academic Talents Support Program for the Young and Middle-Aged.

References

- Hareesh Bahuleyan and Olga Vechtomova. 2017. Uwaterloo at semeval-2017 task 8: Detecting stance towards rumours with topic independent features. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 461–464.
- Qian Chen, Xiaodan Zhu, Zhenhua Ling, Si Wei, Hui Jiang, and Diana Inkpen. 2016. Enhanced lstm for natural language inference. *arXiv preprint arXiv:1609.06038*.
- Leon Derczynski, Kalina Bontcheva, Maria Liakata, Rob Procter, Geraldine Wong Sak Hoi, and Arkaitz Zubiaga. 2017. Semeval-2017 task 8: Rumoureal: Determining rumour veracity and support for rumours. *arXiv preprint arXiv:1704.05972*.
- Omar Enayet and Samhaa R El-Beltagy. 2017. Niletmrgr at semeval-2017 task 8: Determining rumour and veracity support for rumours on twitter. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 470–474.
- William Ferreira and Andreas Vlachos. 2016. Emergent: a novel data-set for stance classification. In *Proceedings of the 2016 conference of the North American chapter of the association for computational linguistics: Human language technologies*, pages 1163–1168.
- Alec Go, Richa Bhayani, and Lei Huang. 2009. Twitter sentiment classification using distant supervision. *CS224N Project Report, Stanford*, 1(12).
- Genevieve Gorrell, Kalina Bontcheva, Leon Derczynski, Elena Kochkina, Maria Liakata, and Arkaitz Zubiaga. 2018. Rumoureal 2019: Determining rumour veracity and support for rumours. *arXiv preprint arXiv:1809.06683*.
- Elena Kochkina, Maria Liakata, and Isabelle Augenstein. 2017. Turing at semeval-2017 task 8: Sequential approach to rumour stance classification with branch-lstm. *arXiv preprint arXiv:1704.07221*.
- Saif Mohammad, Svetlana Kiritchenko, Parinaz Sobhani, Xiaodan Zhu, and Colin Cherry. 2016. Semeval-2016 task 6: Detecting stance in tweets. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 31–41.
- Vahed Qazvinian, Emily Rosengren, Dragomir R Radev, and Qiaozhu Mei. 2011. Rumor has it: Identifying misinformation in microblogs. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 1589–1599. Association for Computational Linguistics.
- Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training. URL https://s3-us-west-2.amazonaws.com/openai-assets/research-covers/languageunsupervised/language_understanding_paper.pdf.
- Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know what you don’t know: Unanswerable questions for squad. *arXiv preprint arXiv:1806.03822*.
- Siva Reddy, Danqi Chen, and Christopher D Manning. 2018. Coqa: A conversational question answering challenge. *arXiv preprint arXiv:1808.07042*.
- Vikram Singh, Sunny Narayan, Md Shad Akhtar, Asif Ekbal, and Pushpak Bhattacharyya. 2017. Iitp at semeval-2017 task 8: A supervised approach for rumour evaluation. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 497–501.
- Feixiang Wang, Man Lan, and Yuanbin Wu. 2017. Ecnu at semeval-2017 task 8: Rumour evaluation using effective features and supervised ensemble models. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 491–496.
- Yukun Zhu, Ryan Kiros, Rich Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *Proceedings of the IEEE international conference on computer vision*, pages 19–27.
- Arkaitz Zubiaga, Maria Liakata, Rob Procter, Geraldine Wong Sak Hoi, and Peter Tolmie. 2016. Analysing how people orient to and spread rumours in social media by looking at conversational threads. *PloS one*, 11(3):e0150989.