CUHK at SemEval-2020 Task 4: CommonSense Explanation, Reasoning and Prediction with Multi-task Learning

Hongru Wang[1], **Xiangru Tang**[3], **Sunny Lai**[1], **Kwong Sak Leung***[1],
Jia Zhu[4], **Gabriel Pui Cheong Fung**[2], **Kam-Fai Wong**[2]
[1]Department of Computer Science and Engineering
The Chinese University of Hong Kong
[2] Department of Systems Engineering and Engineering Management
The Chinese University of Hong Kong
[3]Institute of Computing Technology, Chinese Academy of Sciences
[4]School of Computer Science, South China Normal University
{hrwang, slai, ksleung}@cse.cuhk.edu.hk
jzhu@m.scnu.edu.cn {pcfung, kfwong}@se.cuhk.edu.hk

Abstract

This paper describes our system submitted to task 4 of SemEval 2020: Commonsense Validation and Explanation (ComVE) which consists of three sub-tasks. The challenge is to directly validate whether the system can recognize natural language statements that make sense from those that do not, and also require to generate reasonable explanation. Based on BERT architecture with multi-task setting, we propose an effective and interpretable "Explain, Reason and Predict" (ERP) system to solve the three sub-tasks about commonsense: (a) Validation, and (c) Explanation, (b) Reasoning, following the order of the competition. Inspired by cognitive studies of common sense, our system first generate a reason or understanding of the sentences and then choose which one statement makes sense, which is achieved by multi-task learning. The rational experiment validates our assumption and boost the performance. During the post-evaluation, our system has reached 92.9% accuracy in subtask A (rank 11), 89.7% accuracy in subtask B (rank 8), and BLEU score of 12.9 in subtask C (rank 9)[1].

1 Introduction

How to integrate common sense into natural language models is attracting more and more attention. Common sense, as ordinarily conceived, present themselves as the aspect of the grammar of expressions and sentences on which their semantic properties and relations depend (Asher and Vieu, 1995). And a critical difference of text understanding between humans and machines lies in the fact that humans can access commonsense knowledge while processing text, which helps them draw inferences about effects that are not mentioned in a paragraph. Therefore, it's a fundamental question on how to validate whether a system has a commonsense capability, and more importantly, let the system explain how it reasons using hidden facts. Existing benchmarks measure commonsense knowledge indirectly and without explanation. Also, existing datasets test common sense indirectly through tasks that require extra knowledge, such as co-reference resolution, or reading comprehension. They verify whether a system is equipped with common sense by testing whether it can give a correct answer to make the complete sentence reasonable. However, there are some restrictions on such benchmarks. First, they do not provide a straight quantitatively standard to measure sense masking capability. Second, they do not explicitly identify the key factors required in a sense-making process. And also, they do not need the model to explain why it makes that prediction.

Common sense reasoning require the agent or the model to utilize a world knowledge to take inferences or deep semantic understanding, not only the pattern recognition.

Some empirical analysis has been done previously for common sense reasoning, mainly focus on the form of question answering (QA) (Talmor et al., 2019). But question-answering is hard to directly

[1]All results before 29 April, 2020

*Corresponding author.

Proceedings of the 14th International Workshop on Semantic Evaluation, pages 391–400
Barcelona, Spain (Online), December 12, 2020.

evaluate the commonsense in contextualized representations. And there has been few work investigating commonsense in pre-trained language models (Zhou et al., 2019), such as ELMo (Peters et al., 2018) and BERT (Devlin et al., 2019). Introduced by (Wang et al., 2019a), sense-making is a task to tests whether a model can differentiate sense-making and non-sensemaking statements. Specifically, the statements typically differ only in one keyword which covers nouns, verbs, adjectives, and adverbs. There are two existing approaches that can address this problem, one simple way is to use more commonsense knowledge can be learned from larger training sets (Wang et al., 2019b). On the other hands, some works (Lin et al., 2019) focus on effectively utilizing external, structured commonsense knowledge graphs, such as ConceptNet (Speer et al., 2016) and COMET (Bosselut et al., 2019). Insipred by previous works, more researchers are trying to fuse commonsense knowledge and language model (Forbes and Choi, 2017), and apply them to downstream tasks (Zhong et al., 2019). Recently, a new hybrid approach has been proposed for common sense reasoning (He et al., 2019). The core idea behind it is multi-task learning (Liu et al., 2019b), which has been widely applied in natural language tasks (Liu et al., 2019a).

But existing work in this area has been frustratingly slow, and much of the work is completely theoretical. The field might well benefit if commonsense argumentation were systematically described and evaluated. To tackle it, this system focuses on a benchmark to directly test whether a system can differentiate sentences that make sense from those that do not make sense. Our results indicate that pre-trained models are not able to demonstrate well on the benchmark, and some remaining cases demonstrating that human level is not achieved yet. Thus, we design a new procedure to handle the commonsense challenge inspired by human cognition. It firstly explain its understanding of the given sentences by a language model, and induce the hidden common sense fact. And then, the explanation is used as a supplementary input to the prediction module. Still, we believe that our approach also can be applied to more challenging data sets.

The organization of this paper is as follows: in Section 2, we introduce the basic information about pre-trained language model and task definition. We then describe the framework of our model in Section 3. Empirical results are given and discussed in Section 4. And then we provide more exhaustive analysis for some bad cases that appeared at our experiment in Section 5. Finally, we conclude this survey and in Section 6.

2 Preliminaries

2.1 Task Definition

Formally, the dataset is composed of 10 sentences: two sentences s_1, s_2, three options o_1, o_2, o_3, three references r_1, r_2, r_3. The first two natural language statements that are two similar sentences but only one of them makes sense are used in subtask A called Validation, For the against-common-sense statement s_1 or s_2, we have three optional sentences o_1, o_2 and o_3 to explain why the statement contradicts common sense. The subtask B, named Explanation (Multi-Choice), requires that the model can identify the only correct reason from distractors. Finally, subtask C naming Explanation (Generation), asks the model to generate the reason why it does not make sense. Each statement is paired with three possible explanations r_1, r_2 and r_3 from different perspective(Wang et al., 2020).

Subtask A: Unlike other classification problem, subtask A gives us two statements s_1, s_2 which have similar wordings. Their dependency tree or semantic structure is extremely similar and that requires us to build a model which can recognize these subtle differences and reasoning to judge the sentence whether or not it makes sense.

Subtask B: Subtask B gives us one false sentence s_f(either s_1 or s_2) which means this sentence does not make sense and three options o_1, o_2, o_3. We need to choose one right option which can explain why the give sentence does not make sense.

Subtask C: Subtask C provides one false sentence s_f as same as in subtask B and three references r_1, r_2, r_3. All these three references can explain why the false sentence does not make sense. This task requires us to build a model to generate the correct reason automatically given one false sentence.

2.2 Pretrained Language Model

BERT as a bidirectional pre-trained language model, has recently shown excellent performance in different downstream tasks(Devlin et al., 2019). It is an encoder based on multi-head attention with the self-attention mechanism in a fully connected layer. The input representation of BERT is constructed by summing the corresponding token, segment, and position embeddings. As an autoencoding (AE) model, It can capture the global context in both forward and backward directions. The pre-train of BERT uses two unsupervised strategies. 1) Masked LM; 2) Next Sentence Prediction (NSP). By optimizing for both of two tasks, BERT not only can learn semantic and synthetic knowledge but also world knowledge(Rogers et al., 2020). These explain why BERT has astonishing performance.

RoBERTa is a extended study of BERT which showed that carefully tuning hyper-parameters and increase training data size lead to significantly improved results on language understanding. More specifically, (Liu et al., 2019c) proposed three methods to improve BERT 1) training the model longer, with bigger batches, over more data; 2) removing the next sentence prediction task; 3) training on longer sequences, and 4) dynamically changing the masking pattern applied to the training data. As same as other NLP tasks, RoBERTa gets more higher accuracy compared with BERT.

3 Models

Our proposed ERP system first generates (explain) its understanding of the given sentences by a language model, and then the explanation is used as a supplementary input to the prediction module. For subtask A the input is a sentence pair s_1 and s_2, and the input is the gainst-common-sense statement $s\_$ for subtask B. Subtask C is an explanation generation task and in this way, we could explore common-sense reasoning in two settings 1) explain-and-then-predict and 2) predict-and-then-explain to evaluate the effectiveness of our ERP system. Therefore, we illustrate the ERP system consecutively for different sub-tasks in Sections 3.1 and 3.2.

The architecture of our model is shown in Figure 1, the input x represents sequences (either one sentence or stacked sentences), and then for each token in this sequence is constructed by summing the corresponding token, segment and position embeddings. Then the semantic encoder map the input token into a vector in word-level (token-level), the transformer encoder captures the contextual information in sentence-level via the self-attention mechanism. After we get the contextual embedding vector, we use task-specific layer to apply downstream tasks, we use text classification layer here for both of task A and task B. We choose to introduce subtask A and subtask B first, and followed by subtask C for intuitive understanding, but for the competition, the organizer release the datasets of subtask A, subtask C, and subtask B in turn, which supports our ERP system.

3.1 Sub-task A and Sub-task B

For both of subtask A and subtask B, we cast them as text classification problems. First of all, the training set of subtask A is $\Omega = \left\{(s\_1^1, s\_2^1, y^1), (s\_1^2, s\_2^2, y^2), ..., (s\_1^N, s\_2^N, y^N)\right\}$, in which $s\_$ stands for two similar sentences and y is label. In order to fine-tuning our model, we modified the input sequence x to $x =$ "$[CLS] + s\_1 + [SEP] + s\_2$", the [CLS] token is used for the final classfication, the [SEP] token is used to separate different sentences.

Secondly, for subtask B, during training and validation, the organizer already release the correct explanation for each sent in subtask B, we need to use these data to generate the explanation for test data in subtask B, section 3.2 will introduce more details. Therefore, the generated explanation can be used to improve the performance of our model. As shown in Figure 1, the input sequence consists of one false sent, three options, and some explanations (either ground-truth or generated) according to different periods. The training and validation sample can be cast as $\Omega = \left\{(s^1, a^1, b^1, c^1, e_1^1, e_2^1, e_3^1), (s^2, a^2, b^2, c^2, e_1^2, e_2^2, e_3^2), ..., (s^N, a^N, b^N, c^N, e_1^N, e_2^N, e_3^N)\right\}$, the test samples are $\Omega = \left\{(s^1, a^1, b^1, c^1, e_g^1), (s^2, a^2, b^2, c^2, e_g^2), ..., (s^N, a^N, b^N, c^N, e_g^N)\right\}$. we still use the same structure to arrange our input but use additional special token to disambiguate different functions of sentences, like we use [OPTION] to represent three options, [EXP] to represent explanations. The objective of both

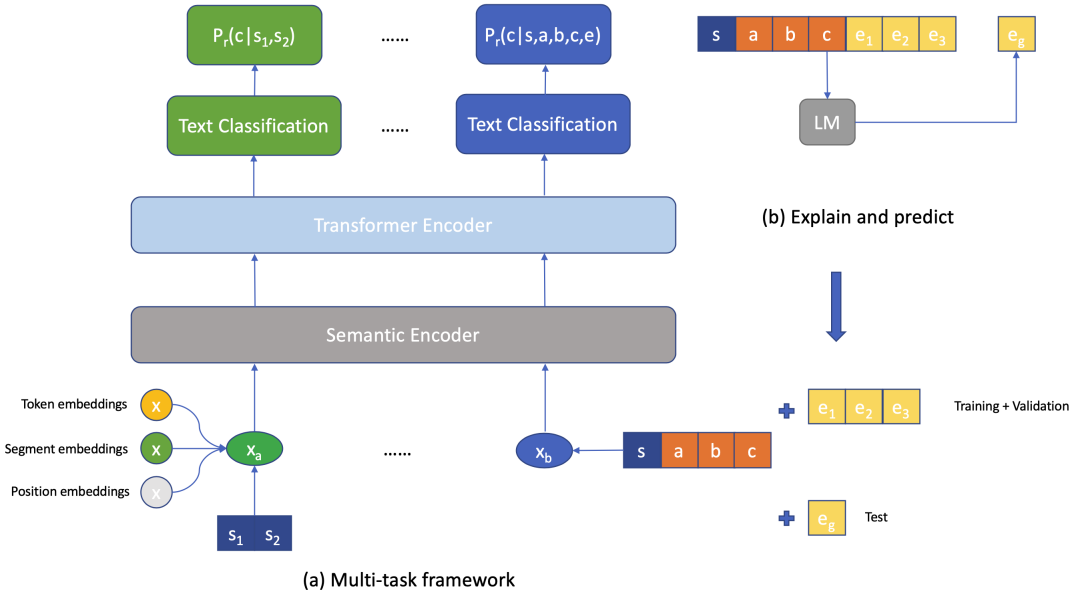

Figure 1: Explain, Reason and Predict (ERP) system, during traing and validation or test, subtask B uses different inputs.

subtask A and subtask B is to maximize:

$$L = \sum_{i=1}^{N} logp(y_i|x, \Theta) \tag{1}$$

3.2 Sub-task C

Here, we employ Commonsense Auto-Generated Explanations in (Wang et al., 2019a), generated by a language model. Subtask C provides one incorrect sentence and three references for explanation. All these three references can explain why the incorrect sentence does not make sense. Our LM is the large, pre-trained OpenAI GPT (Radford et al., 2018), which is a multi-layer transformer (Vaswani et al., 2017) decoder. GPT is fine-tuned on the datasets. Thus, the input contains during the fine-tuning can be described as follows:

$$C_{ans} = \text{“}s\ CUZ\ a, b, c\text{”} \tag{2}$$

where the special token CUZ means “is wrong may because”. the input during testing is defined as follows:

$$C_{ans} = \text{“}s, CUZ\ ?\text{”} \tag{3}$$

The model is trained to generate the explanation e on the basis of conditional language modeling objective, the goal is to maximize:

$$\sum_{i} logP(e_i|e_{i-k}, ..., e_{i-1}, C_{ans}; \theta) \tag{4}$$

where k is the size of the context window. The conditional probability P is modeled by a neural network with parameters Θ conditioned on C_{ans} and previous explanation tokens.

4 Experiment

It is important to make it clear that all our experiments are conducted which meet the requirement of the competition. We can not use the dataset which is not released during the formal competition which means we can not use subtask B data for subtask A and subtask C, because subtask B is released at last, etc.

4.1 Baseline of Sub-task A and Subtask B

As described before, the project consists of three subtasks. Subtask A is to choose from two natural language statements with similar wordings which one makes sense and another one does not make sense. Subtask B is to find the key reason why a given statement does not make sense. Subtask C asks the machine to generate reasons. Subtask A and B are evaluated by accuracy and Subtask C is evaluated using BLEU. The organizers use a random subset of the test set and will do a human evaluation to further evaluate the systems with a relatively high BLEU score (which is not conducted in the post-evaluation period).

First of all, we use BERT and RoBERTa as our baseline since both of them show impressive performance in many NLP downstream tasks. Table 1 shows results compare BERT with RoBERTa that use different corpus for each task. For subtask A, the RoBERTa model reaches the highest accuracy 86.2% in the test and 88.5% in dev datasets. For subtask B, when we add data from subtask A, the performance get the peak at 82.3% accuracy, but it attracts our attention when we use additional subtask C data that the dev accuracy is extremely high with the test accuracy is obviously lower. We assume 1) the data from subtask C show tremendous potential ability to solve subtask B 2) the model relies too much on Subtask C data, resulting in very low performance without it. After we use the generated explanation during the test, the model gets considerable improvement which validates our assumption.

Model	Task	Test	Dev	Label
BERT	Task A	85.3%	85.9%	2
	Task B	79.1%	79.7%	3
	+ Task A	73.4%	82.4%	3
	+ Task C	$51.5\%^1, 54.6\%^2$	82.5%	3
RoBERTa	Task A	**86.2%**	**88.5%**	2
	Task B	81.4%	84.6%	3
	+ Task A	**82.3%**	84.5%	3
	+ Task C	$46.5\%^1, 48.9\%^2$	**99.9%**	3

Table 1: Task A and B: Baseline Results, $51.5\%^1$ represents we do not have explanations during test, but $54.6\%^2$ means we use our generated explanations, etc.

4.2 Explain and Predict

To better understand this deviant phenomenon, we present results with different sample percentages when we randomly choose whether or not to use the subtask C data which is shown at Table 2. Specifically, under the condition of 7:3 sample percent, when we get a sample from subtask B, and then we choose to inject additional explanation with a 30% probability, but not with 70% probability. If we decide to inject additional knowledge, then we will sample one, two, and three explanations with the equal possibility. We observe that the accuracy of dev datasets becomes a little lower, but the test accuracy gets comparable improvement. We force the model to learn more with limited external knowledge through this approach, and the result further validates our assumption before. This leads to the appearance of our "Explain, Reason and Predict (ERP) system and provides an interpretable foundation. Then all we need to do is to improve our baseline, in which we try different ways such as knowledge inject and multi-tasks.

4.3 Multi-task

During the experiment, we find an interesting case illustrating that multi-task learning may help a lot at sub-tasks A and B. Given a false sentence s: an umbrella can help you keep warm in snowy days. and three options: A. we don't wear umbrellas, B: umbrellas can keep you dry in snowy days, C: going outside is very crazy in snowy days. The ground truth is A, but the model outputs B which can better explain the sentence. After we check the other true sentence a thicker cloth can help you keep warm in snowy days in

Sample Percent	Corpus	Test	Dev
5:5	Single Exp	**80.6%**	**89.9%**
	+ Task A	80.5%	86.6%
	All Exp + Task A	80.0%	88.2%
7:3	Single Exp	**82.3%**	**87.1%**
	+ Task A	81.1%	85.5%
	All Exp + Task A	81.9%	86.6%

Table 2: Rational Experiment on Task B, 5:5 means we use 50% of samples to train with exp, 50% to train without exp, etc.

subtask A dataset, we know why the ground truth is A. Obviously, we need some knowledge at subtask A to help us to solve task B, so we think multi-tasks learning is a direction worthy to try (Liu et al., 2019b).

Rather than enriching semantic embedding with knowledge graph, we leverage existing datasets across different domains which also require common sense reasoning like ARC, CommonseQA, and so on. We believe multi-task learning can learn more robust and universal embedding and then make our model get better performance and improve our baseline. In the following experiments, we will validate including additional datasets as external input information can boost our performance of our ERP system.

Task	Model	Test	Dev
Task A	MT-SAN on Task A (single task)	91.7%	94.8%
	MT-SAN on Task A+MNLI+SciTail+MRPC	92.8%	94.8%
	MT-SAN (ensemble)	**92.9%**	**95.1%**
Task B	MT-SAN on Task B (single task)	87.3%	88.1%
	MT-SAN on Task B+ARC+CommonseQA	89.6%, 89.3%*	91.0%, **93.5%***
	MT-SAN (ensemble)	**89.7%**	92.2%

Table 3: Multi-Task Result, $93.5\%^*$ means Explain and Predict we said before.

Table 3 shows the results obtained by our final Multi-task ERP model. We report our two best models that ensemble models using different dropout rates, see more details in the following section. At subtask A, our ensemble model reaches 92.9% accuracy and 95.1% accuracy during test and dev respectively, while getting 89.7% accuracy at the test of subtask B and 93.5% accuracy at dev of subtask B. The highest accuracy in dev dataset of subtask B indicates the tremendous potential of our ERP system. Using additional datasets together to train provides marginal improvement compare with a single task which attributes to better model generalization under multi-task setting from our point of view.

4.4 Implement details

Our implementation of Multi-Task ERP system is based on (Liu et al., 2019b). We used Adamax as our optimizer with a learning rate of 5e-5 and a batch size of 4.We set the number of epochs of 10 and use a linear learning rate decay schedule with warm-up over 0.1. We also set the dropout rate of all the task-specific layers as 0.1, except for ensemble models which we set different dropout rates to get different models. According to (Liu et al., 2019a), we set dropout rate ranged in $\{0.1, 0.2, 0.3\}$. To avoid the exploding gradient problem, we clipped the gradient norm within 1. We set the mixture ratio as 0.4 to re-weighting different tasks(Xu et al., 2018), more details can be found in the implementation [2].

4.5 Subtask C

Since this is a text generation problem, we choose to use the GPT model as our baseline. Since some of the samples use knowledge from subtask A, we conducted contrast experiments by using data from

[2]The codes are available at: https://github.com/ruleGreen/myMTDNN

subtask A and CoS-E(Rajani et al., 2019). We observed that adding explanations led to a very small decrease in the performance compared to the baseline at test datasets, but adding data from subtask A improve about 0.3. Some generated explanations are showed in Table 5 below.

Model	Corpus	Test	Dev
GPT	Task C	12.65	5.96
	+ Task A	**12.94**	5.99
	+ Aug	12.31	**6.54**

Table 4: Task C: CommonSense Explanation

False Sent True Sent	Generated Exp
The inverter was able to power the house The inverter was able to power the house	The inverter was able to power the house
There are beautiful planes here and there in the garden There are beautiful flowers here and there in the garden	planes are not found in the garden.
The chef put extra lemons on the pizza. The chef put extra mushrooms on the pizza.	the chef does not put extra lemons on the pizza.
She is hypnotizing a book. She is reading a book.	books are not hypnotizing.
The boy ate a basketball The boy ate a donut	basketballs are not edible.

Table 5: Task C: Generated Exp(Also used in explain and predict part)

Compared with the original paper(Wang et al., 2019a), our model gets much higher accuracy in both of subtask A and subtask B. Our performance rank 10th on 29 April 2019, with 92.9% accuracy at subtask A (rank 11), 89.7% at subtask B (rank 9), 12.9 at subtask C (rank 8)[3].

5 Analysis

Despite the strong performance of our model, it still fails to detect some samples at subtask A and subtask B, and few sentences generated by our model can not well explain why the given sentence does not make sense. An in-depth analysis of these samples shows that they can be clustered into some classes.

5.1 Error Analysis at Subtask A

- **Basic common sense knowledge** which can be solved by introduce external knowledge graph like ConceptNet (eg., s_1: The moon sets at night, s_2: The sun sets at night, label: 1, prediction: 2).
- **Implicit common sense knowledge.** Current knowledge graphs do not contain everything about common sense knowledge because the limitation of memory and the huge volume of common sense, and it still needs better solutions by using more comprehensive knowledge representation and transfer learning or other methods (eg., s_1: Cats have got seven lives, s_2: Cats have got one life, label: 1, prediction: 2).
- **Specific domain knowledge** required to make a correct judgment (eg., s_1: Hair is already dead, s_2: Hair screams when you cut it, label: 2, prediction: 1), since the human may not know that the hair is dead protein cells, so it is a big challenge for the model to learn this rare domain knowledge from large corpus and datasets.
- **Others** (eg., s_1: coffee takes sleep, s_2: coffee depresses people, label: 2, prediction: 1; s_1: The sun is black, s_2: The sun is white, label: 1, prediction: 2)

[3]All results before 29 April, 2020

5.2 Error Analysis at Subtask B

For subtaskB, there are two different ways to address it: the conventional and the explain and predict methods, that leads three different cases 1) both methods are wrong 2) the conventional one is correct but the other wrong 3) ERP system is correct but the other wrong. According to a comprehensive analysis below, we find that our ERP system can reason and make a more persuasive decision than the conventional one.

1. Both methods output the wrong judgment
   (a) **Explain in different perspectives or levels** (eg., s_f: Everyone loves reading horror novels. o_1: Horror novels are scary. o_2: Reading novels can be a good way to relax. o_3: Not everyone likes to read horror novels. label: C, prediction: A). Why the given s_f does not make sense can have multiple explanations in different levels. Here, to explain why "Everyone loves reading horror novels" does not make sense, from our point of view, both o_1 and o_3 are correct if we assume the given s_f is already false, since they can composite "s_f is wrong because o_1 or o_3". We think o_1 gives explanation from more subtle and deeper level than o_3.
   (b) **Implicit common sense knowledge** as same as in subtask A (eg., s_f: drama plays are often performed before cows, o_1: this rural drama tells the story of a cow, o_2: the cow is a kind of animal while drama isn't, o_3: a cow is unable to appreciate and understand the drama, label: C, prediction: B), we need to know that drama plays are appreciated and understand by people in this example.

2. Examples classified wrongly by the conventional methods but not our ERP system
   (a) **Lack of reasoning capability** which equipped in our ERP system (eg., s_f: shoes can fly, o_1: There are many creatures that can fly, o_2: Shoes do not have wings, o_3: People cannot fly, label: B, prediction: C). The conventional can not reason those wings are needed to fly here.
   (b) **Basic common sense knowledge**. The model still needs external knowledge to support making the right classification.

3. About 1.10% of samples are not classified correctly by our model but the conventional ones, we think this mostly attribute to noise introduced by multi-task setting.
   (a) **Capture plausible knowledge** (eg., s_f: the lava was warm and soft, o_1: lava can destroy the warm and soft cake, o_2: lava is too hard to be soft, o_3: lava is too hot to be warm or soft, label: C, prediction: B). The model captured that something is too hard to be soft, but it ignores the attributes of lava.
   (b) **Others** (eg., s_f: it is said that Santa comes on Thanksgiving Days, o_1: Santa comes on Christmas day, not Thanksgiving Day, o_2: Santa is a figure in legend, not reality, o_3: Santa is a figure in western culture, not eastern culture, label: A, prediction: B). We first think this is caused by explaining in different perspectives or levels which described above, but after we check the whole data, and we find an example (s_f: Santa Claus sent Jim a Christmas present, o_1: There aren't Santa Claus in the world, o_2: Santa Claus is very busy, o_3: Santa Claus is old, label: A) in the training data of subtask B. This proves that our model can learn more robust and universal embedding than the conventional method.

5.3 Error Analysis at Subtask C

Although most of the results make sense, but there are still some generated reasons which can not well explain why the given sentence does not make sense. Most cases, as we found, are with:

- **Wrong explain direction** (eg., s_f: The inverter was able to power the continent, e_g: inverter is not a living thing).
- **Repetition** (eg., s_f: sugar is used to make coffee sour, e_g: sugar is used to make coffee). Like the example, some cases contain repeatedly generated words.

6 Conclusion

In this paper we present our model on the task of Commonsense Validation and Explanation (ComVE) in SemEval-2020. We explore multi-task learning to jointly learn how to inference the hidden common-sense fact and do common-sense reasoning with the RoBERTa model and achieved competitive results. Our result analysis indicates that our Explain, Reason and Predict approach helps improve the performance of RoBERTa and have a strong reasoning capability. The biggest regret in this competition is that we did not incorporated with world knowledge by introducing some knowledge graphs. Due to implicit common sense knowledge and different explanation perspectives, we still need more efforts on the model architecture and find more elegant ways to inject knowledge. Our positive results point to future work in extending the ERP approach to a variety of other types of common sense reasoning tasks.

Acknowledgement

This work is partly supported by Hong Kong RGC GRF (14204118) and ITF (ITS/335/18). We thank the three anonymous reviewers for the insightful suggestions on various aspects of this work.

References

Nicholas Asher and Laure Vieu. 1995. Toward a geometry of common sense: A semantics and a complete axiomatization of mereotopology. In *IJCAI (1)*, pages 846–852.

Antoine Bosselut, Hannah Rashkin, Maarten Sap, Chaitanya Malaviya, Asli elikyilmaz, and Yejin Choi. 2019. Comet: Commonsense transformers for automatic knowledge graph construction. In *ACL*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. *ArXiv*, abs/1810.04805.

Maxwell Forbes and Yejin Choi. 2017. Verb physics: Relative physical knowledge of actions and objects. In *ACL*.

Pengcheng He, Xiaodong Liu, Weizhu Chen, and Jianfeng Gao. 2019. A hybrid neural network model for commonsense reasoning. *ArXiv*, abs/1907.11983.

Bill Yuchen Lin, Xinyue Chen, Jamin Chen, and Xiang Ren. 2019. Kagnet: Knowledge-aware graph networks for commonsense reasoning. In *EMNLP/IJCNLP*.

Xiaodong Liu, Pengcheng He, Weizhu Chen, and Jianfeng Gao. 2019a. Improving multi-task deep neural networks via knowledge distillation for natural language understanding. *ArXiv*, abs/1904.09482.

Xiaodong Liu, Pengcheng He, Weizhu Chen, and Jianfeng Gao. 2019b. Multi-task deep neural networks for natural language understanding. In *ACL*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019c. Roberta: A robustly optimized bert pretraining approach. *ArXiv*, abs/1907.11692.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proc. of NAACL*.

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/researchcovers/languageunsupervised/language understanding paper. pdf*.

Nazneen Fatema Rajani, Bryan McCann, Caiming Xiong, and Richard Socher. 2019. Explain yourself! leveraging language models for commonsense reasoning. In *ACL*.

Anna Rogers, Olga Kovaleva, and Anna Rumshisky. 2020. A primer in bertology: What we know about how bert works. *ArXiv*, abs/2002.12327.

Robyn Speer, Joshua Chin, and Catherine Havasi. 2016. Conceptnet 5.5: An open multilingual graph of general knowledge. In *AAAI*.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. 2019. Commonsenseqa: A question answering challenge targeting commonsense knowledge. *ArXiv*, abs/1811.00937.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Cunxiang Wang, Shuailong Liang, Yue Zhang, Xiaonan Li, and Tian Gao. 2019a. Does it make sense? and why? a pilot study for sense making and explanation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4020–4026, Florence, Italy, July. Association for Computational Linguistics.

Xiaoyan Wang, Pavan Kapanipathi, Ryan Musa, Mo Yu, Kartik Talamadupula, Ibrahim Abdelaziz, Maria Chang, Achille Fokoue, Bassem Makni, Nicholas Mattei, and Michael J. Witbrock. 2019b. Improving natural language inference using external knowledge in the science questions domain. In *AAAI*.

Cunxiang Wang, Shuailong Liang, Yili Jin, Yilong Wang, Xiaodan Zhu, and Yue Zhang. 2020. SemEval-2020 task 4: Commonsense validation and explanation. In *Proceedings of The 14th International Workshop on Semantic Evaluation*. Association for Computational Linguistics.

Yichong Xu, Xiaodong Liu, Yelong Shen, Jingjing Liu, and Jianfeng Gao. 2018. Multi-task learning with sample re-weighting for machine reading comprehension. In *NAACL-HLT*.

Peixiang Zhong, Di. Wang, and Chunyan Miao. 2019. Knowledge-enriched transformer for emotion detection in textual conversations. In *EMNLP/IJCNLP*.

Xuhui Zhou, Yue Zhang, Leyang Cui, and Dandan Huang. 2019. Evaluating commonsense in pre-trained language models. *arXiv preprint arXiv:1911.11931*.