ZJUKLAB at SemEval-2021 Task 4:
Negative Augmentation with Language Model for Reading Comprehension of Abstract Meaning

Xin Xie[1,2][*], **Xiangnan Chen**[1,2][*], **Xiang Chen**[1,2][*], **Yong Wang**[3],
Ningyu Zhang[1,2][†], **Shumin Deng**[1,2], **Huajun Chen**[1,2][†]

[1] Zhejiang University [2] AZFT Joint Lab for Knowledge Engine [3] Microsoft

{xx2020,xnchen2020,xiang_chen}@zju.edu.cn
{zhangningyu,231sm,huajunsir}@zju.edu.cn
wangyon@microsoft.com

Abstract

This paper presents our systems for the three Subtasks of SemEval Task4: Reading Comprehension of Abstract Meaning (ReCAM). We explain the algorithms used to learn our models and the process of tuning the algorithms and selecting the best model. Inspired by the similarity of the ReCAM task and the language pre-training, we propose a simple yet effective technology, namely, negative augmentation with language model. Evaluation results demonstrate the effectiveness of our proposed approach. Our models achieve the **4th** rank on both official test sets of Subtask 1 and Subtask 2 with an accuracy of 87.9% and an accuracy of 92.8%, respectively[1]. We further conduct comprehensive model analysis and observe interesting error cases, which may promote future researches.

1 Introduction

Past decades have witnessed the huge progress of representation learning in Natural Language Processing (NLP). With pre-trained language models, machine reading comprehension (MRC) models can extract answers from given documents and even yield better performance than humans on benchmark datasets such as Squad (Rajpurkar et al., 2016). However, these successes sometimes lead to the hype in which these models are being described as "understanding" language or capturing "meaning" (Bender and Koller, 2020). Note that the intention of MRC is letting the systems read a text like human beings, extracting text information and understanding the meaning of a text then answering questions, which means the systems can not only conclude the semantic of the text but also comprehend the abstract concepts under the constraint of general knowledge regarding

the world (Wang and Jiang, 2016). Nevertheless, little works as well as benchmarks focus on this direction.

SemEval 2021 Task4 (Zheng et al., 2021) is an MRC task that focuses on evaluating the model's ability to understand abstract words. Reading Comprehension of Abstract Meaning (ReCAM) task is divided into three Subtasks including **Subtask 1**: ReCAM-Imperceptibility, **Subtask 2**: ReCAM-Nonspecificity and **Subtask 3**: ReCAM-Intersection. Unlike previous MRC datasets such as CNN/Daily Mail (Hermann et al., 2015), SQuAD (Rajpurkar et al., 2018), and CoQA (Reddy et al., 2019) that request computers to predict concrete concepts, e.g. named entities. This task challenges the model's ability to fill the abstract words removed from human-written summaries based on the English context.

Note that this task's input format is similar to the MLM pre-training task of BERT (Devlin et al., 2019), which aims to predict the mask tokens. Pre-trained language models (PLMs) such as BERT (Devlin et al., 2019), RoBERTa (Liu et al., 2019), ALBERT (Lan et al., 2020), DeBERTa (He et al., 2021) have achieved success on MRC tasks. Inspired by this, we introduce a simple yet effective method, namely, **N**egative **A**ugmentation with **L**anguage model (**NAL**) in **SemEval 2021 Task4**. Specifically, we augment the answer distribution with an additional negative candidate from the mask language model's prediction. Previous work (Petroni et al., 2019; Zhou et al., 2020) indicates that the pre-trained language model has already captured much world knowledge. Thus, we argue that knowledge can help guild the model training and identify those ambiguous abstract meanings. Further, we introduce other technologies such as label smoothing, domain-adaptive pre-training in our system. We describe the detailed approaches used for the Subtasks in Section 3.

---

[*] Equal contribution and shared co-first authorship.
[1] Our implementation is publicly available at https://github.com/zjunlp/SemEval2021Task4

We conduct comprehensive experiments in Section 3, and we achieve the 4th system for Subtask 1: ReCAM-Imperceptibility and the 4th system for Subtask 2: ReCAM-Nonspecificity in the leaderboard. In our experiments, we observe that PLMs without fine-tuning can easily get 60+% accuracy on both Subtask 1 and Subtask 2, demonstrating that pre-trained language models already capture some abstract meanings. We further find that our negative augmentation with language model can improve the performance with **2.6%** in Subtask 1 and **4.6%** in Subtask 2. Finally, we conduct error analysis to promote future researches.

2 Background

Machine reading comprehension (MRC) has received increasing attention recently, which is a challenging task. According to the type of the answer, reading comprehension tasks can be divided into four categories (Chen, 2018): 1) Cloze-style: The question contains a "@placeholder," and the system must choose a word or entity from the set of candidate answers to fill in the "@placeholder" to make the sentence complete. 2) Multiple choice: In this type of task, Choosing a suitable answer from K sets of given answers. This answer can be one word or a sentence. 3) Span prediction: This kind of task is also called (Extractive question answering), which requires the system to extract a suitable range of text fragments from a given original text based on the question as to the answer. 4) Free-form answer: This task allows the answer to be any type of text, which is necessary to mine deep-level contextual semantic information according to a given question and a collection of candidate documents, and even combine multiple articles to give the best answer.

In **SemEval 2021 Task4**, it requires the system to have a strong ability of reading comprehension not only because the task is the cloze-style format as mentioned above but also the abstract words in answers. There are two definitions of abstract words: imperceptibility and nonspecificity. Concrete words refer to things, events, and properties that we can perceive directly with our senses (Spreen and Schulz, 1966; Turney et al., 2011). Compared to concrete words like "trees" and "red," abstract words for imperceptibility are created by humans instead of pointing the things in the natural world. For example, as shown in Table 1, "want" and "achieve" means a person's attitude to-

P:	Briton Davies won F42 shot put gold with a Games record at Rio 2016, but was unable to defend his 2012 discus title as it did not feature in Brazil. "I don't normally say what I'm going for," said the Welshman, 25. "But this time *I'm definitely going for the two golds in both disciplines* and nothing will be better than being in front of a home crowd." ...
Q:	Paralympic champion Aled Sion Davies @placeholder two gold medals at the 2017 World Para Athletics Championships in London.
A:	(A) suffered (B) promoted (C) remains **(D) wants** (E) achieved
P:	... Low vitamin D levels can lead to brittle bones and rickets in children. The figures from the HSCNI show a dramatic rise in Vitamin D prescriptions over the last 10 years: The data does not include Vitamin D bought over the counter...
Q:	Rickets does not have the ring of a 21st Century problem - it sounds more like the @placeholder of a bygone era .
A:	**(A) horror** (B) size (C) fate (D) tale (E) death

Table 1: Examples of the **SemEval 2021 Task 4**. Given a passage and a question, the model needs to pick the best one from the five candidates to replace @placeholder.

wards something and a person's accomplishment about something. Meanwhile, the abstract words for nonspecificity can be described as upper words. By determining whether one word can generalize another word, we can get dictionaries of different levels. The words with higher levels are the nonspecificity words. Compared to concrete concepts like groundhog and whale, hypernyms such as vertebrate are regarded as more abstract (Changizi, 2008).

The difference between Subtask 1 and Subtask 2 is the definition of abstract words. So the input of both Subtask 1 and Subtask 2 are the same. The input of these tasks are shown in Table 1, it can be represented as a triple $< P, Q, A >$, where $P = s_1, s_2, ..., s_m$ is the passage from CNN daily (Hermann et al., 2015), Q is a human-written summary based on the passage with one abstract word replaced by "@placeholder" and A is a set of can-

didate abstract words for filling in the "@placeholder" in the question.

3 System Overview

3.1 Model Design

Recently, with the development of the large Pretrained Language Models (PLMs), such as GPT (Radford et al., 2018), BERT (Devlin et al., 2019), RoBERTa (Liu et al., 2019), ALBERT (Lan et al., 2020), DeBERTa (He et al., 2021), have overwhelm the NLP community (Zhang et al., 2020c). The powerful semantic feature extraction capabilities of the PLMs make us only need to make better use of the BERT-like model itself for downstream tasks instead of adding different layers to the model.

Similar to the normal multi-choice task, we have five candidates, one passage, and one question per sample. Here we leverage PLMs as encoders to capture the global context representation about the passage, question, and answer. Then a decoder is used to determine the score of each $< P, Q, A >$ pair. Since we get $A_1, ..., A_n$ n answers, for every passage, we construct n input samples as $[Q - A_i; P]$, the concatenation of $Q - A_i$ and P. Because the question is the summary with an abstract word removed. We construct $Q - A$ by replacing "@placeholder" with the option from the candidate set instead of concatenating Q and A. After encoding all n inputs for a single passage, we get the global representations T_i for different options in the candidate set. During fine-tuning PLMs, the first special token [CLS] represents the global meaning of the whole input. We use an dense decoder layer to compute the score for all T_i, the calculation of score is as follow:

$$T_i = PLM(Q - A; P) \tag{1}$$

$$score_i = \frac{\exp\left(f\left(T_i\right)\right)}{\sum_{i'} \exp\left(f\left(T_{i'}\right)\right)} \tag{2}$$

where the $[Q - A; P]$ is the input constructed according to the instruction of PLMs and MRC tasks, and the T_* is the final hidden state of the first token [CLS]. The candidate answers with higher *scores* will be identified as the final prediction.

Since previous research (Gao et al., 2020; Yang et al., 2019) demonstrate that there exists a gap between language model pre-training and fine-tuning
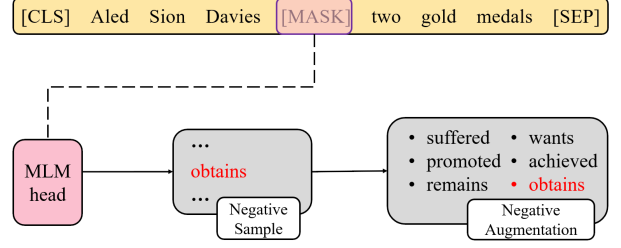

Figure 1: The procedure of Negative Augmentation with Language Model (NAL).

the models in the downstream task and inspire by the similar task definition as MLM, we introduce the negative augmentation with language model mechanism (Section 3.2). Note that the additional label will enhance the discriminability of the abstract meanings in a contrastive manner. In other words, the model is encouraged **NOT** to generate those abstract tokens from the language model, but the golden candidates from the given documents. We further introduce the label smoothing (Section 3.3), which can enhance the model performance. Finally, we leverage task-adaptive pre-training (Section 3.4) inspired by (Gururangan et al., 2020) to obtain better performance.

3.2 Negative Augmentation with Language Model

Inspired by the same format of MLM and this task, we first conduct a toy experiment to test whether a PLM can get the right answer without any supervised signal. Firstly we replace the "@placeholder" with [MASK] to reconstruct the input and ask the BERT model with MLM head to predict the word token at the [MASK]. Then we calculate the similarity between the word model predict and the options from the set of candidate answers. We set the option with the highest similarity score as the model's choice. Then we find that the BERT model without any fine-tuning gets **60+%** accuracy in both Subtask 1 and Subtask 2. The result above shows that PLMs have the ability to predict abstract words, and those predicted words can be leveraged as negative candidates in the fine-tuning period.

Note that huge languages have quantities of parameters; the PLMs are able to store much knowledge through pre-training tasks. However, [MASK] is not used when fine-tuning the model for downstream tasks; how to use the knowledge stored by the model on pre-training tasks more explicitly on downstream tasks has become a hot
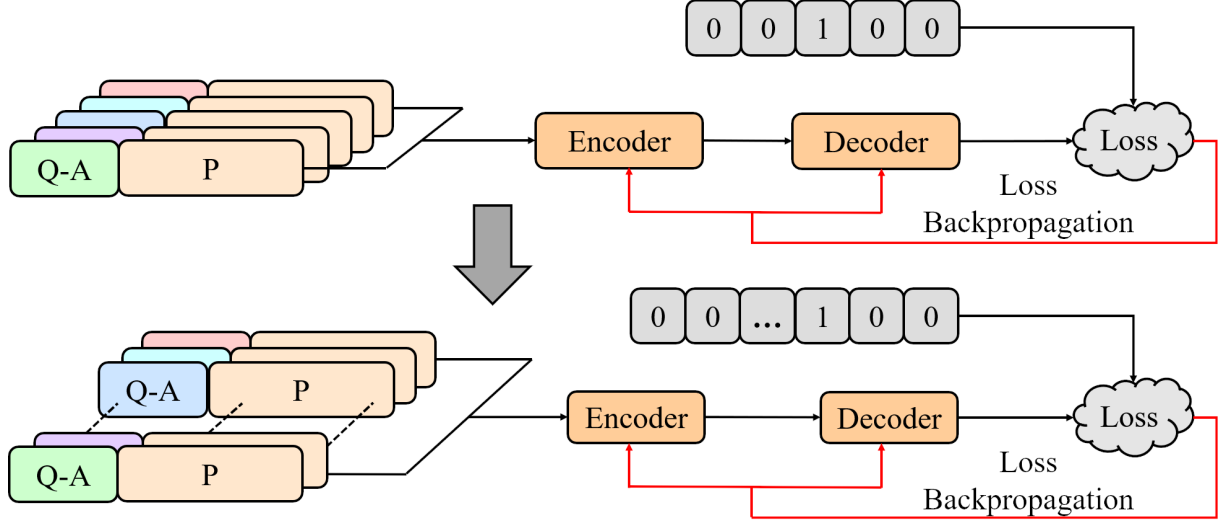
Figure 2: System overview (Best viewed in color.). The top of the Figure refers to the normal fine-tuning of multi-choice models, ignoring the form of pre-training tasks. While the bottom of the Figure refers to our system with Negative Augmentation with Language Model (NAL), which uses the abstract words predict by the original PLM as negative candidates to augment fine-tuning.

topic of current research. Motivated by this, we try to bridge the gap between pre-train and downstream tasks. Inspired by the contrastive learning (Chen et al., 2020; Robinson et al., 2020) as stronger negative samples will help the model learning with better performance, we introduce our negative augmentation with language model method. Specifically, we let the PLMs predict the "@placeholder" replaced with [MASK] token to generate negative candidates. Thus, we can leverage those negative words that may mislead the models to help train the models. Formally, we have:

$$P = p(m_i|\theta, [Q-A; P]), m_i \in [1, 2, ..., |V|] \quad (3)$$

where P are the distribution of words model, predict, m_i is the token in the vocabulary, and $|V|$ is the total number of the vocabulary. We can use the distribution to get the top confusing words to augment our models, which is described in Figure 2. Due to the limitation of GPU, we add the possible word to augment our models.

3.3 Label Smoothing

Label smoothing is a well-known "trick" to improve the model's performance effectively. It encourages the activations of the penultimate layer to be close to the template of the correct class and equally distant to the templates of the incorrect classes (Müller et al., 2019). With more options

than the original dataset by the approach mentioned in Section 3.2, label smoothing will magnify our method's effect while fine-tuning the models. Suppose the output of the final layer and softmax layer as follows:

$$p_k = \frac{e^{x^T w_k}}{\sum_{l=1}^{L} e^{x^T w_l}} \quad (4)$$

where p_k is the likelihood the model assigns to the k-th class, w_k represents the weights and biases of the last layer. x is the vector containing the activations of the penultimate layer of a neural network concatenated with "1" to account for the bias. let us see the equitation about the cross entropy loss.

$$L = -\sum_{c=1}^{M} y_k \log (p_k) \quad (5)$$

The cross-entropy formula without Label smoothing only focuses on whether the positive example is true and does not pay attention to the negative examples' relationship. We make the soft y as follows:

$$y_i = \begin{cases} (1 - \varepsilon), & \text{right answer} \\ \frac{\varepsilon}{K-1}, & \text{wrong answer} \end{cases} \quad (6)$$

We set ε as 0.1 in our models.

3.4 Task-Adaptive Pre-training

The BERT-like model is pre-trained in the general domain corpus such as Wikipedia. Since passages

Statistics / Dataset	Subtask 1	Subtask 2
# Train	3,227	3,318
# Trail	1,000	1,000
# Dev	837	851
# Test	2,025	2,017
Avg. # Length Per Passage	262	418

Table 2: Statistics of the **SemEval 2021 Task 4** dataset.

mainly come from CNN daily, the data distribution may be quite different from pre-training data. Therefore, we utilize task-adaptive pre-train BERT with masked language model and next sentence prediction tasks on the domain-specific data. Task-adaptive pre-training not only makes the model better fit the distribution in the domain but also helps the model to predict good negative words to enhance the original dataset, which is described in Section 3.2. We take two different approaches for task-adaptive pre-training as follows:

1) In-domain pre-training, we use the source data: CNN Daily to task-adaptive pre-training our base models(Sun et al., 2020).

2) Within-task pre-training, practically we replace the "@placeholder" with the correct answer and put the same input format as the fine-tuning steps, which is $[Q - A; P]$ (Gururangan et al., 2020).

4 Experimental Setup

4.1 Dataset

In Subtask 1, the training/trail/development/test contains $3,227/1,000/837/2,025$ instances. In Subtask 2, the training/trail/development/test contains $3,318/1,000/851/2,017$ instances. The overall statistics can be found in Table 2.

4.2 Pre-processing

For data pre-processing, we use the byte-level BPE encoding (Sennrich et al., 2016), and the official vocabulary contains more than fifty thousand byte-level tokens. All tokens are stored in MERGES.TXT, while VOCAB.JSON is a byte-to-index mapping. Generally speaking, the higher the frequency, the smaller the byte index. Since the average length of the passage about Subtask 1 and Subtask 2 is 262 and 418, we divide those long context paragraphs. We limit the max number of tokens in an input sample $[Q - A; P]$ to 256 for our system. Statically, 60% of the paragraphs exceeds

the 256 tokens (including the special tokens like [CLS], [SEP] and so on. For these input samples, we divide them into new input samples with at most 256 tokens. To be more specific, we divide the passage to different inputs with the same question and answer.

4.3 Hyper-parameter Setting

Our system is implemented with PyTorch (Paszke et al., 2019) and we use the PyTorch version of the pre-trained language models[2]. We employ RoBERTa, ALBERT, and DeBERTa large models as our PLM encoder. We use AdamW optimizer (Loshchilov and Hutter, 2018) to fine-tune the models. We set the batch size to 1, and the max length of input to 256 for RoBERTa, 128 for ALBERT.

Usually, the batch size has a significant influence on the BERT-like model; due to the limit of GPU memory, we use gradient accumulation in our training steps. We set the gradient accumulation step as 32, which means the formal number of batch sizes is 32 in training. We pick the best learning rate from the dev set, fine-tuning the RoBERTa, ALBERT, DeBERTa with the learning rate of 9×10^{-6}, 1×10^{-5} and 1×10^{-5} respectively. We set the number of epoch to 8 for ALBERT and 12 for RoBERTa and DeBERTa. Furthermore, we save the best model on the validation set for testing during training. Because the formats of both Subtask 1 and Subtask 2 are the same, we set the same batch size and max length of the input sequence for training.

5 Results

5.1 Subtask 1 Results

On Subtask 1 , the ReCAM-Imperceptibility task, the evaluation results are illustrated in Table 3. We set the three baseline models: RoBERTa$_{Large}$, DeBERTa$_{Large}$, and ALBERT$_{xxLarge}$. RoBERTa$_{Large}$ + NAL, DeBERTa$_{Large}$ + NAL, and ALBERT$_{Large}$ + NAL denotes the language model with our proposed negative augmentation with language model. Ensemble refers to the ensemble model of the three models as mentioned above with all strategies. We find that ALBERT achieves better performance in Subtask 1 but fails to get good performance in Subtask 2, while DeBERTa and RoBERTa have better performance in Subtask 2.

---

[2] https://github.com/huggingface/transformers (version 3.3.0)

Model	Dev	Test
Baseline		
RoBERTa$_{Large}$	83.3	-
ALBERT$_{xxLarge}$	85.1	-
DeBERTa$_{Large}$	84.1	-
Ours		
RoBERTa$_{Large}$ + NAL	85.9	86.1
ALBERT$_{xxLarge}$+ NAL	86.2	85.6
DeBERTa$_{Large}$+ NAL	86.7	86.8
Ensemble	**88.5**	**87.9**

Table 3: Results (Accuracy) on Subtask 1.

Model	Dev	Test
Baseline		
RoBERTa$_{Large}$	86.7	-
ALBERT$_{xxLarge}$	84.3	-
DeBERTa$_{Large}$	87.7	-
Ours		
RoBERTa$_{Large}$ + NAL	91.1	89.7
ALBERT$_{xxLarge}$+ NAL	89.3	88.6
DeBERTa$_{Large}$+ NAL	91.3	90.3
Ensemble	**93.7**	**92.8**

Table 4: Results (Accuracy) on Subtask 2.

Comparing with the original RoBERTa, DeBERTa, and ALBERT models, each model is hugely improved with NAL by about **2.1**% accuracy. We further observe that DeBERTa and RoBERTa, which have the same architecture, obtain better performance than ALBERT in the dev and test sets. We think the possible reason is that ALBERT uses layer weight sharing, which reduces the model's generalization ability in reading comprehension, especially the abstract words meaning. Finally, the ensemble of the best model of RoBERTa, DeBERTa, and ALBERT lead to a significant improvement (**4.3**% accuracy) compared with baselines, which is also our final submission to the leaderboard.

5.2 Subtask 2 Results

On Subtask 2, the ReCAM-Nonspecificity task, the experiment results are showed in Table 4. Similar to the models in Subtask 1, we choose RoBERTa, DeBERTa and ALBERT as our baseline models. All RoBERTa$_{Large}$ + NAL , ALBERT$_{xxLarge}$ + NAL and DeBERTa$_{Large}$ + NAL are the models with negative augmentation with

language model. Ensemble refers to the ensemble model of RoBERTa, DeBERTa, and ALBERT with all strategies. We notice that our proposed mechanism brings significant improvement (averaging **4.3**% of the accuracy score) compared with baselines, demonstrating the effectiveness of our proposed strategies such as negative augmentation with a language model, label smoothing, and task-adaptive pre-training. We observe that ensemble approach of three enhanced models (RoBERTa$_{Large}$ + NAL, ALBERT$_{xxLarge}$+ NAL and DeBERTa$_{Large}$+ NAL) obtain the best accuracy of **92.8**% at test set, which is also our final submit to the leaderboard.

5.3 Subtask3 Results

Subtask3 focuses on the model's transferability. During the evaluation period, we use the data on Subtask 2 to evaluate the models trained on the Subtask 1 and vice versa. We obtain the 82% accuracy of the model trained on Subtask 1 and evaluated on Subtask 2 on the dev set.

During experiments for all tasks, we have tried to use different decoders like MLP and other network architecture. Eventually, we find that it does not help to improve the system's performance. An explanation is that the pre-trained language models (PLMs) have already captured global contextual sentence meaning at the `[CLS]` token.

5.4 Further Analysis

5.4.1 Analysis of Negative Augmentation with Language Model

During our experiments, we conduct case studies to figure out how our method of NAL helps the model to boost performance. From Table 5, we notice that the original PLM considers using the "all", "half" as its choice instead of "parts". Although fine-tuned on the downstream task, the baseline model still choose "half". In our NAL method, we add some misleading negative words to help models correct the knowledge learned from the pre-training task.

5.4.2 Analysis of Passage Length

In usual MRC tasks, the length of the passage is a key factor for the models to solve the problems. We conduct experiments to analyze the performance regarding different lengths of passage. Contrary to the common assumption, from Figure 3 and Figure 4, we observe that the instances with long passage obtain better performance. We think

Example	
Question:	The Aurora Borealis, better known as the Northern Lights, was spotted across @placeholder of England on Sunday.
Answer set	{(A) millions, **(B) parts**, (C) half, (D) isle, (E) remains}
NAL set	{all, half, **parts**}
Baseline	(C) half
Model with NAL	(B) parts
Question:	The BBC is providing live coverage of the Scottish National Party conference in Glasgow. This live @placeholder has finished .
Answer set	{(A) results, (B) recording, **(C) event**, (D) action, (E) center}
NAL set	{blog, recording , stream}
Baseline	(B) recording
Model with NAL	(C) event

Table 5: We can clearly see the negative options can help the model better understand the abstract meaning in the passage and question. Answers are **bold** in the Table.

that abstract mean understanding may need comprehensive context information from the long sentence, and we will conduct further analysis in future works.
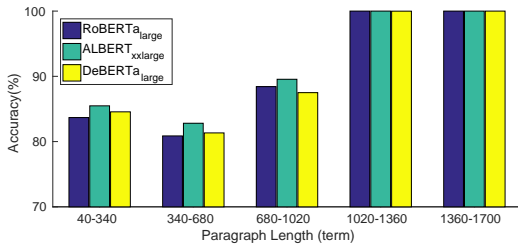

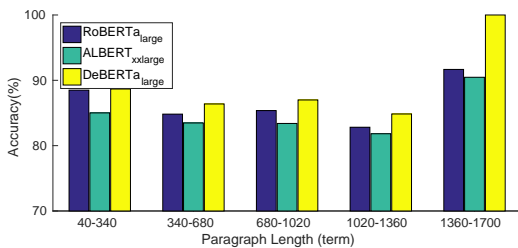Figure 3: Results (Accuracy) on Subtask 1 with the length of passage.

Figure 4: Results (Accuracy) on Subtask 2 with the length of passage.

5.4.3 Case Study

We select four kinds of different types of error cases to promote further researches. We classify the examples according to the main causes (pretraining, fine-tuning, and so on) of the error. We think it will help us better understand what the model learns from pre-training and fine-tuning.

Case 1 - Influenced by the original pre-training task

- **Passage:** "...found the United States to have the highest number of sleep deprived students, with 73% of 9 and 10 year olds and 80% of 13 and 14 year olds identified by their teachers as *being adversely affected*. The BBC's Jane O'Brien reports."
- **Question:** Sleep deprivation is a significant hidden factor in lowering the @placeholder of school pupils , according to researchers carrying out international education tests .
- **Answer:** (A) morale (B) IQ (C) mortality (D) closure (E) achievement
- **Negative augmented choice:** (F) intelligence
- **Right Option:** (E) achievement
- **Wrong Option:** (B) IQ
- **Potential causes:** After pre-training on the large general domain corpus, PLMs have a huge bias on predicting the `[MASK]` token. Just like the "IQ" model predict in the "@placeholder". Even after fine-tuning, our models still cannot recognize the strong evidence "being adversely affected". In our daily life, we wouldn't hold that being adversely affected by lack of sleep can lead to a decrease in IQ. We usually say that lack of sleeping may lower one's achievement in the future.
- **How to help models?** To prevent the model from relying too much on pre-training tasks, we create more negative samples to help the model to understand what is wrong or right

about the abstract words.

Case 2 - Adverse affected by fine-tuning

- **Passage:** ” 17 May 2017 Last updated at 12:44 BST Adrien Gulfo, wearing red, who plays for the Swiss side Pully Football, tried to clear the ball away from his goal with a spectacular bicycle kick. Unfortunately for him it all went very wrong - watch the video... There was a happy ending to the story for Gulfo though, Pully went through to the cup final on penalties after the match finished 3-3.”
- **Question:** You won't believe this own goal that was @placeholder in the Swiss lower league !
- **Answer:** (A) scored (B) born (C) eliminated (D) closed (E) beaten
- **Negative Augmented Choice:** (F) scored (model predict ”scored.” Because it is the right answer, so we choose another choice ”played” as an augmented choice.)
- **Right Option:** (A) scored
- **Wrong Option:** (E) beaten
- **Potential Causes:** It is quite weird that the original PLMs can predict the right answer, but fail to make it after fine-tuning. We suppose that in the process of fine-tuning, the inconsistency of abstract vocabulary prediction and the interference of other vocabulary caused the model's effect in some cases to decrease instead.
- **How to help models?** We could use our approach of NAL to increase the weight of the knowledge learned in the pre-training task or leverage external knowledge (Zhang et al., 2019, 2020b; Yu et al., 2020; Zhang et al., 2020a).

Case 3 - Obscure abstract word meaning

- **Passage:** ” ...Mr Habgood said: ”We're pretty sure it will be popular because it was when East Street was closed for other reasons and we want to make it a friendlier place to be. ”It does fit with our larger objectives to improve the town and make it safer for cyclists and pedestrians.” ...”
- **Question:** Three busy town center streets are to be pedestrianised in a bid to improve @placeholder for shoppers and cyclists .
- **Answer:** (A) opportunities (B) services (C) quality (D) disruption (E) safety

- **Negative Augmented Choice:** (F) access
- **Right Option:** (E) safety
- **Wrong Option:** (B) services
- **Potential Causes:** Due the limit of GPU memory, we cannot put the long passage into the model once a time. So during the training, the model can only see a small chunk of the passage, so that it cannot get the global representation of the passage.
- **How to help models?** We chunk those long sentences with the approach of the sliding window to help the model understanding the whole passage.

Case 4 - Hypernyms is not always right

- **Passage:** ” North Wales Fire and Rescue Service was called to Express Linen Services on Vale Road in Llandudno Junction just before 19:30 GMT on Thursday. North Wales Police said a man was treated at the scene for smoke inhalation. Police have asked people to avoid the area...”
- **Question:** A number of @placeholder have been evacuated as firefighters tackle a blaze at a commercial laundry firm 's premises in Conwy county.
- **Answer:** (A) families (B) properties (C) water (D) disruption (E) vehicles
- **Negative Augmented Choice:** (F) homes
- **Right Option:** (B) properties
- **Wrong Option:** (A) families
- **Potential Causes:** Hypernyms is the main focus of Subtask 2, the model may consider the ”families” as the upper level of the ”people” occur in the passage and choose the ”(A) families” instead of the right answer ”(B) properties”.
- **How to help models?** We try to use the proposed NAL to add more abstract words learned from the pre-training to mitigate this issue.

6 Conclusion

This paper presents our system design for the SemEval 2021 Task4. We propose a simple yet effective method called negative augmentation with language model. Comprehensive experiments demonstrate the effectiveness of our proposed approach. We also conduct case studies and investigate why the model fails to obtain the correct prediction.

Note that language models are pre-trained from the huge corpus; recently, researchers have iden-

tified the bias in the language model, which may mislead the model prediction. Our proposed negative augmentation with language model can help the model better discriminate candidates in fine-tuning, thus boost the performance. From another perspective, as depicts in Section 3.2, the language model without any fine-tuning gets **60+%** accuracy in both Subtask 1 and Subtask 2. This indicates that bias exists in the datasets (Part of the abstract meaning can be obtained from the language model). More strong benchmarks should be constructed in the future.

References

Emily M. Bender and Alexander Koller. 2020. Climbing towards NLU: On meaning, form, and understanding in the age of data. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5185–5198, Online. Association for Computational Linguistics.

Mark A. Changizi. 2008. Economically organized hierarchies in WordNet and the Oxford English Dictionary. *Cognitive Systems Research*, 9(3):214–228.

Danqi Chen. 2018. *Neural Reading Comprehension and Beyond*. Ph.D. thesis, Stanford University.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey E. Hinton. 2020. A simple framework for contrastive learning of visual representations. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pages 1597–1607. PMLR.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Tianyu Gao, Adam Fisch, and Danqi Chen. 2020. Making pre-trained language models better few-shot learners. *CoRR*, abs/2012.15723.

Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A. Smith. 2020. Don't stop pretraining: Adapt language models to domains and tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8342–8360, Online. Association for Computational Linguistics.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2021. DeBERTa: Decoding-enhanced BERT with Disentangled Attention. *arXiv:2006.03654 [cs]*. ArXiv: 2006.03654.

Karl Moritz Hermann, Tomás Kociský, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. In *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pages 1693–1701.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. ALBERT: A lite BERT for self-supervised learning of language representations. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

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

Ilya Loshchilov and Frank Hutter. 2018. Fixing Weight Decay Regularization in Adam.

Rafael Müller, Simon Kornblith, and Geoffrey E. Hinton. 2019. When does label smoothing help? In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 4696–4705.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 8024–8035.

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. 2019. Language models as knowledge bases? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2463–2473, Hong Kong, China. Association for Computational Linguistics.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training.

Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know what you don't know: Unanswerable questions for SQuAD. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 784–789, Melbourne, Australia. Association for Computational Linguistics.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas. Association for Computational Linguistics.

Siva Reddy, Danqi Chen, and Christopher D. Manning. 2019. Coqa: A conversational question answering challenge. *Trans. Assoc. Comput. Linguistics*, 7:249–266.

Joshua Robinson, Ching-Yao Chuang, Suvrit Sra, and Stefanie Jegelka. 2020. Contrastive learning with hard negative samples. *CoRR*, abs/2010.04592.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Otfried Spreen and Rudolph W. Schulz. 1966. Parameters of abstraction, meaningfulness, and pronunciability for 329 nouns. *Journal of Verbal Learning and Verbal Behavior*, 5(5):459–468.

Chi Sun, Xipeng Qiu, Yige Xu, and Xuanjing Huang. 2020. How to Fine-Tune BERT for Text Classification? *arXiv:1905.05583 [cs]*. ArXiv: 1905.05583.

Peter Turney, Yair Neuman, Dan Assaf, and Yohai Cohen. 2011. Literal and metaphorical sense identification through concrete and abstract context. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 680–690, Edinburgh, Scotland, UK. Association for Computational Linguistics.

Shuohang Wang and Jing Jiang. 2016. Machine comprehension using match-lstm and answer pointer. *CoRR*, abs/1608.07905.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime G. Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 5754–5764.

Haiyang Yu, Ningyu Zhang, Shumin Deng, Hongbin Ye, Wei Zhang, and Huajun Chen. 2020. Bridging text and knowledge with multi-prototype embedding for few-shot relational triple extraction. In *Proceedings of the 28th International Conference on Computational Linguistics, COLING 2020, Barcelona, Spain (Online), December 8-13, 2020*, pages 6399–6410. International Committee on Computational Linguistics.

Ningyu Zhang, Shumin Deng, Juan Li, Xi Chen, Wei Zhang, and Huajun Chen. 2020a. Summarizing chinese medical answer with graph convolution networks and question-focused dual attention. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings, EMNLP 2020, Online Event, 16-20 November 2020*, pages 15–24. Association for Computational Linguistics.

Ningyu Zhang, Shumin Deng, Zhanlin Sun, Jiaoyan Chen, Wei Zhang, and Huajun Chen. 2020b. Relation adversarial network for low resource knowledge graph completion. In *WWW '20: The Web Conference 2020, Taipei, Taiwan, April 20-24, 2020*, pages 1–12. ACM / IW3C2.

Ningyu Zhang, Shumin Deng, Zhanlin Sun, Guanying Wang, Xi Chen, Wei Zhang, and Huajun Chen. 2019. Long-tail relation extraction via knowledge graph embeddings and graph convolution networks. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3016–3025, Minneapolis, Minnesota. Association for Computational Linguistics.

Ningyu Zhang, Qianghuai Jia, Kangping Yin, Liang Dong, Feng Gao, and Nengwei Hua. 2020c. Conceptualized representation learning for chinese biomedical text mining. *arXiv preprint arXiv:2008.10813*.

Boyuan Zheng, Xiaoyu Yang, Yuping Ruan, Quan Liu, Zhen-Hua Ling, Si Wei, and Xiaodan Zhu. 2021. SemEval-2021 task 4: Reading comprehension of abstract meaning. In *Proceedings of the 15th International Workshop on Semantic Evaluation (SemEval-2021)*.

Xuhui Zhou, Yue Zhang, Leyang Cui, and Dandan Huang. 2020. Evaluating commonsense in pretrained language models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 9733–9740.