Implementation Details about Baselines

1. IR Solver

We filter out paragraphs that do not have at least one non-stopword overlap between (question or scenario) and the options. The score of option is the sum score of the remaining top-5 paragraphs.

2. AR

For AR model, we chose C3 and Dureader Dataset to train the AVD (Answer Verifier Discriminator) and DRD (Document Relevance Discriminator) respectively. Specifically, for each question in Dureader, we regarded the most related paragraph of the document which is selected to extract the answer as positive samples, and regarded paragraphs from documents not selected to extract the answer as negative samples. Meanwhile, we limited the negative samples from the same question to no more than 10 for the sake of positive-negative balance. Finally we got 20000 samples as train set and 2000 samples as development and test set. For C3 dataset, we regarded the concatenation of question and the right answer as positive samples and wrong answer as negatives samples. We trained these models for 10 epochs with a learning rate of 3e-5 and batch size of 16. The max sequence length is restricted to be at most 512 tokens long. Finally, we got an accuracy of 0.904, 0.915 for the modified Dureader dataset and 0.6904, 0.7233 for the modified C3 dataset, with Bert-wwm-ext and ERNIE respectively. For the learning to rank phrase of AR, we tune the number of paragraphs involved in rank in {10, 20, 50, 100, 200}, and we achieve best devlopment set score when select 20 paragraph.

3. DPR

For dense retrieval model, we first pre-train it with external data, and then fine-tune it on the target data. We use the Chinese reading comprehension dataset DuReader as the external data and transform it into the ODQA task dataset, in which the paragraph following each question is used as its positive paragraph and the other paragraphs are used as negative paragraphs, and BM25 is used to get the hard negative paragraphs. We tuned maximum sequence length in{128,256},set epochs= 40, and other hyperparameters cording to the authors.

For the reader model, we follow the author's idea of combining paragraph selection and adapt appropriately to adapt the multiple-choice QA task. Specifically, for the top k paragraphs retrieved, first, each option is concatenated with it and input to the encoder to obtain k semantic vectors represented by the special symbol [CLS]. Second, we use the linear layer to convert each semantic vector into an option sub-score. After that, in order to get the complete option score, we need to calculate the paragraph selection score. We concatenate the semantic representation of the same paragraph under multiple options, and input a linear layer to get

the paragraph selection score. Finally, using the softmax function, we weight and sum the sub-scores of the options into the option scores. The option with the highest option score is used as the answer selected by the model. During training, we calculate two losses: paragraph selection and answering questions, which are supervised by paragraph labels and answer labels respectively. For paragraph selection loss, we sample 1 positive paragraph and m-1 negative paragraphs for each question. We tuned m between 2 and 10, and choose m=5 based on the performance of the development set. For questions that can't construct a positive paragraph, we also use it to train the reader, but the paragraph selection loss is not calculated.

4.MMM

For MMM model, following the original paper, we firstly fine-tuned ERNIE and Bert-wwm-ext on a Chinese NLI dataset which is CMNLI in this case. We trained the model for 6 epoches with a learning rate of 1e-5 and batch size of 64. The max sequence length is set to 128. Finally we got an accuracy of 0.8048, 0.8048 for Bert-wwm-ext and ERNIE respectively. For the multi-task stage, we fine-tuned the model obtained from the first stage on a large in-domain source dataset C3 with its its train set and the three target datasets via multi-task learning. The code is adapted from the official code base of https://github.com/jind11/MMM-MCQA.

5. DCMN+

We emailed the author of DCMN+ to get a code. We did not make any special changes to the code.

5. SG-Net

For SG-Net model, we used HanLP as the syntactic dependency parsing tool to make the SDOI (Syntactic Dependency of Intertest) mask. Specifically, we used the "LARGE_ALBERT_BASE" tokenizer, "CTB5_POS_RNN_FASTTEXT_ZH" POS tagger and "CTB7_BIAFFINE_DEP_ZH" syntactic parser. The code is adapted from the official code base of https://github.com/cooelf/SG-Net.

6. DHC

For DHC model, we used 12 attention heads and set the dimension of query, key and value to 768 as Bert-wwm-ext and Ernie did.