# ElimiNet: A Model for Eliminating Options for Reading Comprehension with Multiple Choice Questions

**Soham Parikh***, **Ananya B. Sai***, **Preksha Nema***, **Mitesh M. Khapra**
Indian Institute of Technology, Madras
{sohamp,ananyasb,preksha,miteshk}@cse.iitm.ac.in

## Abstract

The task of Reading Comprehension with Multiple Choice Questions, requires a human (or machine) to read a given {*passage, question*} pair and select one of the $n$ given options. The current state of the art model for this task first computes a question-aware representation for the passage and then *selects* the option which has the maximum similarity with this representation. However, when humans perform this task they do not just focus on option selection but use a combination of *elimination* and *selection*. Specifically, a human would first try to eliminate the most irrelevant option and then read the passage again in the light of this new information (and perhaps ignore portions corresponding to the eliminated option). This process could be repeated multiple times till the reader is finally ready to select the correct option. We propose *ElimiNet*, a neural network-based model which tries to mimic this process. Specifically, it has gates which decide whether an option can be eliminated given the {*passage, question*} pair and if so it tries to make the passage representation orthogonal to this eliminated option (akin to ignoring portions of the passage corresponding to the eliminated option). The model makes multiple rounds of partial elimination to refine the passage representation and finally uses a selection module to pick the best option. We evaluate our model on the recently released large scale RACE dataset and show that it outperforms the current state of the art model on 7 out of the 13 question types in this dataset. Further, we show that taking an ensemble of our *elimination-selection* based method with a *selection* based method gives us an improvement of 3.1% over the best-reported performance on this dataset.

## 1 Introduction

Reading comprehension is the task of answering questions pertaining to a given passage. An AI agent which can display such capabilities would be useful in a wide variety of commercial applications such as answering questions from financial reports of a company, troubleshooting using product manuals, answering general knowledge questions from Wikipedia documents, *etc*. Given its widespread applicability, several variants of this task have been studied in the literature. For example, given a passage and a question, the answer could either (i) match some span in the passage or (ii) be generated from the passage or (iii) be one of the $n$ given candidate answers. The last variant is typically used in various high school, middle school, and competitive examinations. We refer to this as Reading Comprehension with Multiple Choice Questions (RC-MCQ). There is an increasing interest in building AI agents with deep language understanding capabilities which can perform at par with humans on such competitive tests. For example, recently [Lai *et al.*, 2017] have released a large scale dataset for RC-MCQ collected from high school and middle school English examinations in China comprising of approximately 28000 passages and 100000 questions. The large size of this dataset makes it possible to train and evaluate complex neural network based models and measure the scientific progress on RC-MCQ. While answering such Multiple Choice Questions (MCQs) (*e.g.*, Figure 1), humans typically use a combination of *option elimination* and *option selection*. More specifically, it makes sense to first try to eliminate options which are com-

Passage: One day, I was studying at home. Suddenly, there was a loud noise...A building in my neighborhood was on fire...A few people jumped out of the window... Those who were still on the second floor were just crying for help...Firefighters arrived at last. They fought the fire bravely. Water pipes were used and a ladder was put near the second-floor window. Then the people inside were taken out by the firefighters...Thanks to the firefighters, the people inside were saved and the fire was put out in the end, but many things, such as desk, pictures and clothes, were damaged.
***Question:*** *How did the people who didn't jump out of the window get out of the building?*
**Option A:** They were taken out by the firefighters.
**Option B:** They climbed down a ladder by themselves.
**Option C:** They walked out after the fire was put out.
**Option D:** They were taken out by doctors
**Correct Option:** A

Figure 1: Example of RC-MCQ from RACE dataset

*denotes equal contribution

pletely irrelevant to the given question. While doing so, we may also be able to discard certain portions of the passage which are not relevant to the question (because they revolve around the option which has been eliminated, *e.g.*, portions marked in blue and orange, corresponding to Option B and Option C respectively in Figure 1). This process can then be repeated multiple times, each time eliminating an option and refining the passage (by discarding irrelevant portions). Finally, when it is no longer possible to eliminate any option, we can pick the best option from the remaining options. In contrast, the current state of the art models for RC-MCQ focus explicitly on option selection. Specifically, given a question and a passage, they first compute a question aware representation of the passage (say $d_q$). They then compute a representation for each of the $n$ options and select an option whose representation is closest to $d_q$. There is no iterative process where options get eliminated and the representation of the passage gets refined in the light of this elimination.

We propose a model which tries to mimic the human process of answering MCQs. Similar to the existing state of the art method [Dhingra *et al.*, 2017], we first compute a question-aware representation of the passage (which essentially tries to retain portions of the passage which are only relevant to the question). We then use an elimination gate (depending on the passage, question and option) which takes a soft decision as to whether an option needs to be eliminated or not. Next, akin to the human process described above, we would like to discard portions of the passage representation which are aligned with this eliminated option. We do this by subtracting the component of the passage representation along the option representation (similar to Gram-Schmidt orthogonalization). The amount of orthogonalization depends on the soft decision given by the elimination gate. We repeat this process multiple times, during each pass doing a soft elimination of the options and refining the passage representation. At the end of a few passes, we expect the passage representation to be orthogonal (hence dissimilar) to the irrelevant options. Finally, we use a selection module to select the option which is most similar to the refined passage representation. We refer to this model as *ElimiNet*. Note that such a model will not make sense in cases where the options are highly related. For example, if the question is about life stages of a butterfly and the options are four different orderings of the words *butterfly, egg, pupa, caterpillar* then it does not make sense to orthogonalize the passage representation to the incorrect option representations. However, the dataset that we focus on in this work does not contain questions which have such permuted options.

We evaluate *ElimiNet* on the RACE dataset and compare it with Gated Attention Reader (GAR) [Dhingra *et al.*, 2017], the current state of the art model on this dataset. We show that of the 13 question types in this dataset our model outperforms GAR on 7 question types. We also visualize the soft elimination probabilities learnt by *ElimiNet* and observe that it indeed learns to iteratively refine the passage representation and push the probability mass towards the correct option. Finally, we show that an ensemble model combining *ElimiNet* with *GAR* gives an accuracy of $47.2\%$ which is $3.1\%$ (relative) better than the best-reported performance on this dataset.The code for our model is publicly available[1].

## 2 Related Work

Over the last few years, the availability of large scale datasets has led to an increasing interest in the task of Reading Comprehension. These datasets cover different variations of the Reading comprehension task. For example, SQuAD [Rajpurkar *et al.*, 2016], TriviaQA [Joshi *et al.*, 2017], NewsQA [Trischler *et al.*, 2016], MS MARCO [Nguyen *et al.*, 2016], NarrativeQA [Kociský *et al.*, 2017], *etc.* contain {*passage, question, answer*} where the answer matches a span of the passage or it has to be generated. On the other hand, CNN/Daily Mail [Hermann *et al.*, 2015], Children's Book Test (CBT) [Hill *et al.*, 2015] and Who Did What (WDW) dataset [Onishi *et al.*, 2016] offer cloze-style RC where the task is to predict a missing word/entity (from the passage) in the question. Some other datasets such as MCTest [Richardson *et al.*, 2013], AI2 [Khashabi *et al.*, 2016] and RACE contain RC with multiple choice questions (RC-MCQ) where the task is to select the right answer.

The advent of these datasets and the general success of deep learning for various NLP tasks, has led to a proliferation of neural network based models for RC. For example, the models proposed in [Xiong *et al.*, 2016; Seo *et al.*, 2016; Wang *et al.*, 2017; Hu *et al.*, 2017] address the first variant of RC requiring span prediction as in the SQuAD dataset. Similarly, the models proposed in [Chen *et al.*, 2016; Kadlec *et al.*, 2016; Cui *et al.*, 2017; Dhingra *et al.*, 2017] address the second variant of RC requiring cloze-style QA. Finally, [Lai *et al.*, 2017] adapt the the models proposed in [Chen *et al.*, 2016; Dhingra *et al.*, 2017] for cloze-style RC and use them to address the problem of RC-MCQ. Irrespective of which of the three variants of RC they address, these models use a very similar framework. Specifically, these models contain components for (i) encoding the passage (ii) encoding the question (iii) capturing interactions between the question and the passage (iv) capturing interactions between question and the options (for MCQ) (v) making multiple passes over the passage and (vi) a decoder to predict/generate/select an answer. The differences between the models arise from the specific choice of the encoder, decoder, interaction functions and iteration mechanism. Most of the current state of the art models can be seen as special instantiations of the above framework.

The key difference between our model and existing models for RC-MCQ is that we introduce components for (soft-)eliminating irrelevant options and refining the passage representation in the light of this elimination. The passage representation thus refined over multiple (soft-)elimination rounds is then used for selecting the most relevant option. To the best of our knowledge, this is the first model which introduces the idea of option elimination for RC-MCQ.

## 3 Proposed Model

Given a passage $D = [w_1^d, w_2^d, \ldots, w_M^d]$ of word-length $M$, a question $Q = [w_1^q, w_2^q, \ldots, w_N^q]$ of word-length $N$ and $n$ options $Z_k = [w_1^z, w_2^z, \ldots, w_{J_k}^z]$ where $1 \leqslant k \leqslant n$

[1] https://github.com/sohamparikh94/ElimiNet

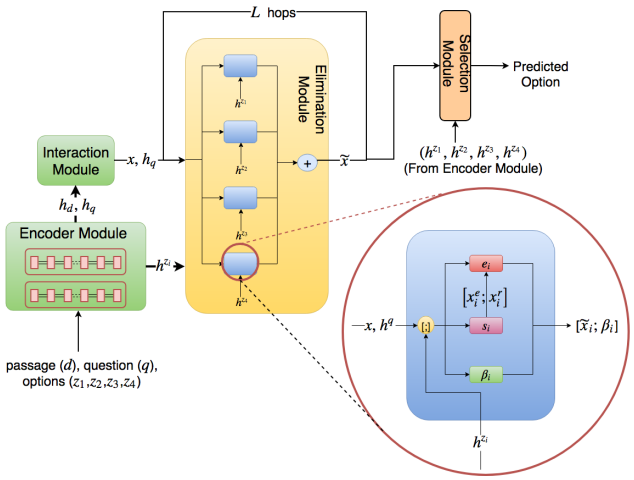


Figure 2: A simplistic diagram of the proposed model

and each option is of word-length $J_k$, the task is to predict a conditional probability distribution over the options (*i.e.*, to predict $P(Z_i|D,Q)$). We model this distribution using a neural network which contains modules for encoding the passage/question/options, capturing the interactions between them, eliminating options and finally selecting the correct option. We refer to these as the encoder, interaction, elimination and selection modules as shown in Figure 2. Among these, the main contribution of our work is the introduction of a module for elimination. Specifically, we introduce a module to (i) decide whether an option can be eliminated (ii) refine the passage representation to account for eliminated/uneliminated options and (iii) repeat this process multiple times. In the remainder of this section, we describe the various components of our model.

**Encoder Module:** We first compute vectorial representations of the question and options. We do so by using a bidirectional recurrent neural network which contains two Gated Recurrent Units (GRU) [Chung *et al.*, 2014], one which reads the given string (question or option) from left to right and the other which reads the string from right to left. For example, given the question $Q = [w_1^q, w_2^q, \ldots, w_N^q]$, each GRU unit computes a hidden representation for each time-step (word) as:

$$\overrightarrow{h_i^q} = \overrightarrow{GRU_q}(\overrightarrow{h_{i-1}^q}, e(w_i^q))$$
$$\overleftarrow{h_i^q} = \overleftarrow{GRU_q}(\overleftarrow{h_{i-1}^q}, e(w_i^q))$$

where $e(w_i^q) \in \mathbb{R}^d$ is the $d$-dimensional embedding of the question word $w_i^q$. The final representation of each question word is a concatenation of the forward and backward representations (*i.e.*, $h_i^q = [\overleftarrow{h_i^q}, \overrightarrow{h_i^q}]$). Similarly, we compute the bi-directional representations for each word in each of the $k$ options as $h_i^{z_k} = [\overleftarrow{h_i^{z_k}}, \overrightarrow{h_i^{z_k}}]$. Just to be clear, $h_i^{z_k}$ is the representation of the $i$-th word in the $k$-th option ($z_k$). We use separate GRU cells for the question and options, with the same GRU cell being used for all the $n$ options. Note that the encoder also computes a representation of each passage word as simply the word embedding of the passage word (*i.e.*, $h_i^d = e(w_i^d)$). Later on in the interaction module we use a GRU cell to compute the interactions between the passage words.

**Interaction Module:** Once the basic question and passage word representations have been computed, the idea is to allow them to interact so that the passage words' representations can be refined in the light of the question words' representations. This is similar to how humans first independently read the passage and the question and then read the passage multiple times, trying to focus on the portions which are relevant and ignoring portions that are irrelevant (*e.g.*, portion marked in red in Figure 1) to the question. To achieve this, we use the same multi-hop architecture for iteratively refining passage representations as proposed in Gated Attention Reader [Dhingra *et al.*, 2017]. At each hop $t$, we use the following set of equations to compute this refinement:

$$\alpha_i^t = \text{softmax}(Q^T d_i^t)$$

where, $Q \in \mathbb{R}^{N \times l}$ is a matrix whose columns are $h_1^q, h_2^q, ..., h_N^q$ as computed by the encoder. $\alpha_i^t \in \mathbb{R}^N$ such that each element $j$ of $\alpha_i^t$ essentially computes the importance of the $j$-th question word for the $i$-th passage word during hop $t$. At the 0-th hop, $d_i^0 = h_i^d = e(w_i^d) \in \mathbb{R}^l$ is simply the embedding of the $i$-th passage word. The goal is to refine this embedding over each hop based on interactions with the question. Next, we compute,

$$\tilde{q}_i^t = Q\alpha_i^t$$

where $\tilde{q}_i^t \in \mathbb{R}^l$ computes the importance of each dimension of the current passage word representation and is then used as a gate to scale up or scale down different dimensions of the passage word representation.

$$\tilde{d}_i^t = d_i^t \odot \tilde{q}_i^t$$

We now allow these refined passage word representations to interact with each other using a bi-directional recurrent neural network to compute $d_i^{(t+1)}$ for the next hop.

$$\overrightarrow{d_i^{(t+1)}} = \overrightarrow{\text{GRU}}_D^{(t+1)}(\overrightarrow{d_{i-1}^{(t+1)}}, \tilde{d}_i^{(t)})$$
$$\overleftarrow{d_i^{(t+1)}} = \overleftarrow{\text{GRU}}_D^{(t+1)}(\overleftarrow{d_{i-1}^{(t+1)}}, \tilde{d}_i^{(t)})$$
$$d_i^{(t+1)} = [\overleftarrow{d_i^{(t+1)}}, \overrightarrow{d_i^{(t+1)}}]$$

The above process is repeated for $T$ hops wherein each hop takes $d_i^{(t)}, Q$ as the input and computes a refined representation $\tilde{d}_i^{(t+1)}$. After $T$ hops, we obtain a fixed-length vector representation of the passage by combining the passage word representations using a weighted sum.

$$m_i = \text{softmax}(\tilde{d}_i^{(T)} W_{att} h_N^q)$$
$$x = \sum_{i=1}^{M} m_i \tilde{d}_i^{(T)} \quad (1)$$

where $m_i$ computes the importance of each passage word and $x$ is a weighted sum of the passage representations.

**Elimination Module:** The aim of the elimination module is to refine the passage representation so that it does not focus on portions which correspond to irrelevant options. To do so we first need to decide whether an option can be eliminated or not and then ensure that the passage representation gets modified accordingly. For the first part, we introduce an *elimination* gate to enable a soft-elimination.

$$e_i = \text{sigmoid}(W_e x + V_e h^q + U_e h^{z_i})$$

Note that this gate is computed separately for each option $i$. In particular, it depends on the final state of the bidirectional option GRU ($h^{z_i} = h^{z_i}_{J_i}$). It also depends on the final state of the bidirectional question GRU ($h^q = h^q_N$) and the refined passage representation ($x$) computed by the interaction module. $W_e, V_e, U_e$ are parameters which will be learned.

Based on the above soft-elimination, we want to now refine the passage representation. For this, we compute $x^e_i$ which is the component of the passage representation ($x$) orthogonal to the option representation ($h^{z_i}$) and $x^r_i$ which is the component of the passage representation along the option representation.

$$r_i = \frac{< x, h^{z_i} > h^{z_i}}{|x|^2}$$

$$x^e_i = x - r_i \quad (2)$$

$$x^r_i = x - x^e_i \quad (3)$$

The *elimination gate* then decides how much of $x^e_i$ and $x^r_i$ need to be retained.

$$\tilde{x}_i = e_i \odot x^e_i + (1 - e_i) \odot x^r_i$$

If $e_i = 1$ (eliminate, *e.g.*, portions corresponding to Option D in Figure 1) then the passage representation will be made orthogonal to the option representation (akin to ignoring portions of the passage relevant to the option) and $e_i = 0$ (don't eliminate, *e.g.*, portions marked in green, corresponding to Option A in Figure 1) then the passage representation will be aligned with the option representation (akin to focusing on portions of the passage relevant to the option).

Note that in equations (2) and (3) we completely subtract the components along or orthogonal to the option representation. We wanted to give the model some flexibility to decide how much of this component to subtract. To do this we introduce another gate, called the subtract gate,

$$s_i = \text{sigmoid}(W_s x + V_s h^q + U_s h^{z_i})$$

where $W_s, V_s, U_s$ are parameters that need to be learned. We then replace the RHS of Equations 2 and 3 by $x - s_i \odot r_i$ and $x - s_i \odot x^e_i$ respectively. Thus the components $r_i$ and $r^\perp_i$ used in Equation (2) and (3) are gated using $s_i$. One could argue that $e_i$ itself could encode this information but empirically we found that separating these two functionalities (elimination and subtraction) works better.

For each of the $n$ options, we independently compute representations $\tilde{x}_1, \tilde{x}_2, ..., \tilde{x}_n$. These are combined to obtain a single refined representation for the passage.

$$b_i = v^T_b \tanh(W_b \tilde{x}_i + U_b h^{z_i})$$

$$\beta_i = \text{softmax}(b_i)$$

$$\tilde{x} = \sum_{i=1}^{n} \beta_i \tilde{x}_i \quad (4)$$

Note that $\tilde{x}_1, \tilde{x}_2, ..., \tilde{x}_n$ represent the $n$ option-specific passage representations and $\beta_i$'s give us a way of combining these option specific representations into a single passage representation. We repeat the above process for $L$ hops wherein the $m$-th hop takes $\tilde{x}^{m-1}$, $h^q$ and $h^{z_i}$ as input and returns a refined $\tilde{x}^m$ computed using the above set of equations.

**Selection Module** Finally, the selection module takes the refined passage representation $\tilde{x}^L$ after $L$ elimination hops and computes its bilinear similarity with each option representation.

$$\text{score}(i) = \tilde{x}^L W_{att} h^{z_i}$$

where $\tilde{x}^L$ and $h^{z_i}$ are vectors and $W_{att}$ is a matrix which needs to be learned. We select the option which gives the highest score as computed above. We train the model using the cross entropy loss by normalizing the above scores (using softmax) first to obtain a probability distribution.

## 4 Experimental Setup

In this section, we describe the dataset used for evaluation, the hyperparameters of our model, training procedure and state of the art models used for comparison.

**Dataset:** We evaluate our model on the RACE dataset which contains multiple choice questions collected from high school and middle school English examinations in China. The high school portion of the dataset (RACE-H) contains 62445, 3451 and 3498 questions in train, validation, and test sets respectively. The middle school portion of the dataset (RACE-M) contains 18728, 1021 and 1045 questions for train, validation, and test sets respectively.
This dataset contains a wide variety of questions of varying degrees of complexity. For example, some questions ask for the most appropriate title for the passage which requires deep language understanding capabilities to comprehend the entire passage. There are some questions which ask for the meaning of a specific term or phrase in the context of the passage. Similarly, there are some questions which ask for the key idea in the passage. Finally, there are some standard Wh-type questions. Given this wide variety of questions, we wanted to see if there are specific types of questions for which an elimination module makes more sense. To do so, with the help of in-house annotators, we categorize the questions in the test dataset into the following 13 categories using scripts with manually defined rules: (i) 6 Wh-question types, (ii) questions asking for the title/meaning/key idea of the passage, (iii) questions asking whether the given statement is True/False, (iv) questions asking for a quantity (e.g., how much, how many) (v) fill-in-the-blanks questions. We were able to classify 91.26% of questions in the test set into these 12 categories and the remaining 8.74% of questions were labeled as miscellaneous. The distribution of questions belonging to each of these categories in RACE-H and RACE-M are shown in Figure 3.

**Training Procedures:** We try two different ways of training the model. In the first case, we train the parameters of all

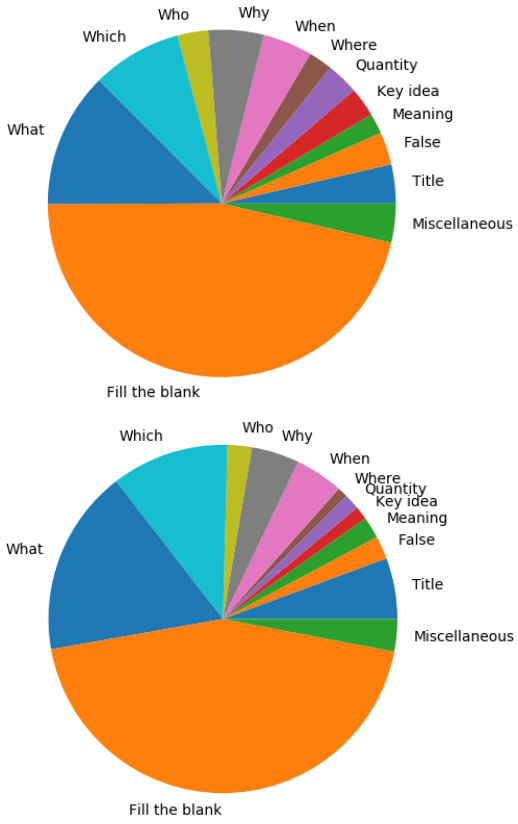


Figure 3: Distribution of different question types in the RACE-Mid (top) and RACE-High (bottom) portions of the dataset

the modules (encoder, interaction, elimination, and selection) together. In the second case, we first remove the elimination module and train the parameters of the remaining modules. We then fix the parameters of the encoder and interaction module and train only the elimination and selection module. The idea was to first help the model understand the document better and later focus on elimination of options (in other words, ensure that the entire learning is focused on the elimination module). Of course, we also had to learn the parameters of the selection module from scratch because it now needs to work with the refined passage representations. Empirically, we find that this pre-training step does not improve over the performance obtained by end-to-end training. Hence, we report results only for the first case (*i.e.*, end-to-end training).

**Hyperparameters:** We restrict our vocabulary to the top 50K words appearing in the passage, question, and options in the dataset. We use the same vocabulary for the passage, question, and options. We use the same train, valid, test splits as provided by the authors. We tune all our models based on the accuracy achieved on the validation set. We initialize the word embeddings with 100 dimensional Glove embeddings [Pennington *et al.*, 2014]. We experiment with both fine-tuning and not fine-tuning these word embeddings. The hidden size for BiGRU is the same across the passage, question, and option and we consider the following sizes :$\{64, 128, 256\}$. We experiment with $\{1, 2, 3\}$ hops in the interaction module and $\{1, 3, 6\}$ passes in the elimination module. We add dropout at the input layer to the BiGRUs and experiment with dropout values of $\{0.2, 0.3, 0.5\}$. We try both Adam and SGD as the optimizer. For Adam, we set the learning rate to $10^{-3}$ and for SGD we try learning rates of $\{0.1, 0.3, 0.5\}$. In general, we find that Adam converges much faster. We train all our models for upto 50 epochs as we do not see any benefit of training beyond 50 epochs.

**Models Compared:** We compare our results with the current state of the art model on RACE dataset, namely, Gated Attention Reader [Dhingra *et al.*, 2017]. This model was initially proposed for cloze-style RC and is, in fact, the current state of the art model for cloze-style RC. The authors of RACE dataset adapt this model for RC-MCQ by replacing the output layer with a layer which computes the bilinear similarity between the option and passage representations.

## 5 Results and Discussions

In this section, we discuss the results of our experiments.

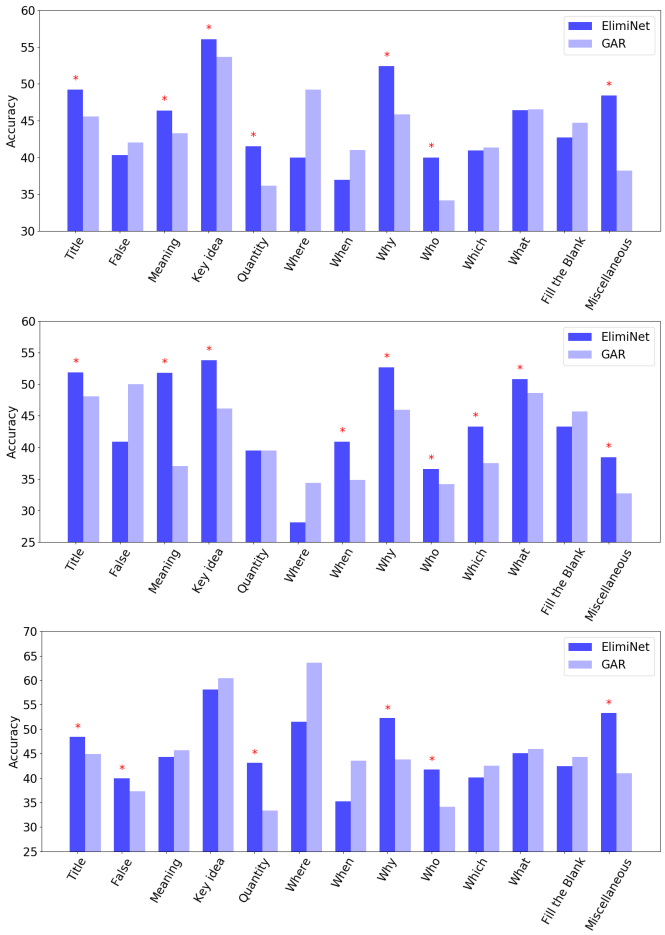


Figure 4: Performance of ElimiNet and Gated Attention Reader (GAR) on different question categories in RACE-Full (top), RACE-Mid (mid) and RACE-High (bottom). The categories in which our model outperforms GAR are marked with *.

### 5.1 Performance of Individual Models

We compare the accuracy of different models on RACE-Mid (middle school), RACE-High (high school) and full RACE test-set comprising of both RACE-Mid and RACE-High. For each dataset, we compare the accuracy for each question type. These results are summarized in Figure 4. We observe that on RACE-Mid ElimiNet performs better than Gated Attention Reader (GAR) on 9 out of 13 categories. Similarly, on RACE-High ElimiNet performs better than GAR on 6 out of 13 categories. Finally, on RACE-full, ElimiNet performs better than GAR on 7 out of 13 categories. Note that, overall on the entire test set (combining all question types) our model gives a slight improvement over GAR. The main reason for this is that the dataset is dominated by fill in the blank style questions and our model performs worse by only $2\%$ on such questions. However, since nearly $50\%$ of the questions in the dataset are fill in the blank style questions even a small drop in the performance on these questions, offsets the gains that we get on other question types.

### 5.2 Ensemble of Different Models

Since ElimiNet and GAR perform well on different question types we believe that taking an ensemble of these models should lead to an improvement in the overall performance. For a fair comparison, we also want to see the performance when we independently take an ensemble of $n$ GAR models and $n$ ElimiNet models. We refer to these as GAR-ensemble and ElimiNet-ensemble models. Each model in the ensemble is trained using a different hyperparameter setting and we use $n = 6$ (we do not see any benefit of using $n > 6$). The results of these experiments are summarized in Table 1. ElimiNet-ensemble performs better than GAR-ensemble and the final ensemble gives the best results. We observe the ElimiNet-ensemble performs significantly better on RACE-Mid dataset than the GAR-ensemble and gives almost the same performance on the RACE-High dataset. Overall, by taking an ensemble of the two models we get an accuracy of $47.2\%$ which is $3.1\%$ (relative) better than GAR and $1.3\%$ (relative) better than GAR-ensemble.

### 5.3 Effect of Subtract Gate

We wanted to see if the subtract gate enables the model to learn better (by performing partial orthogonalization/alignment). For this, we compared the accuracy with and without the subtract gate (we set the subtract gate to a vector of 1s). We observed that the accuracy of our model drops from $44.33\%$ to $42.58\%$ and we outperformed the GAR model only in 3 out of 13 categories. This indicates that the flexibility offered by the subtract gate does help the model.

### 5.4 Visualizing Shift in Probability Scores

If the elimination module is indeed learning to eliminate options and align/orthogonalize the passage representation w.r.t the uneliminated/eliminated options then we should see a shift in the probability scores as we do multiple passes of elimination. To visualize this, in Figure 5, we plot the probabilities of the correct option and the incorrect option with the highest probability before passing through *elimination* module for two different test instances. We observe that as we do multiple passes of elimination, the probability mass shifts from the incorrect option (blue curve) to the correct option (green curve). This indicates that the elimination module is learning to align the passage representation with the correct option (hence, increasing its similarity) and moves it away from the incorrect option (hence, decreasing its similarity).

| Model | RACE-Mid | RACE-High | RACE-Full |
|---|---|---|---|
| SA Reader | 44.2 | 43.0 | 43.3 |
| GA Reader (GAR) | 43.7 | 44.2 | 44.1 |
| **ElimiNet** | **44.4** | **44.5** | **44.5** |
| GAR Ensemble | 45.7 | 46.2 | 45.9 |
| ElimiNet Ensemble | **47.7** | 46.1 | 46.5 |
| GAR + ElimiNet (ensemble of above 2 ensembles) | 47.4 | **47.4** | **47.2** |

Table 1: Performance of individual and ensemble models

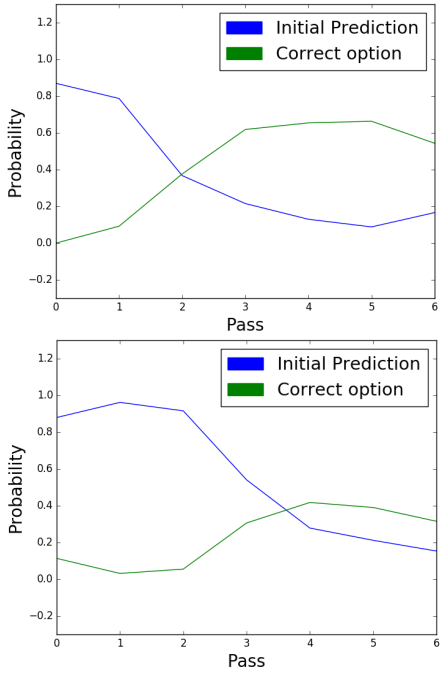


Figure 5: Change in the probability of correct option and incorrect option (initially predicted with highest score) over multiple passes of the *elimination* module. The two figures correspond to two different examples from the test set.

## 6 Conclusion

We focus on the task of Reading Comprehension with Multiple Choice Questions and propose a model which mimics how humans approach this task. Specifically, the model uses a combination of elimination and selection to arrive at the correct option. This is achieved by introducing an elimination module which takes a soft decision as to whether an option should be eliminated or not. It then modifies the passage representation to either align it with uneliminated options or orthogonalize it to eliminated options. The amount of orthog-

onalization or alignment is determined by two gating functions. This process is repeated multiple times to iteratively refine the passage representation. We evaluate our model on the recently released RACE dataset and show that it outperforms current state of the art models on 7 out of 13 question types. Finally, using an ensemble of our elimination-selection approach with a state of the art selection approach, we get an improvement of $3.1\%$ over the best reported performance on RACE dataset. As future work, instead of soft elimination we would like to use reinforcement learning techniques to learn a policy for hard elimination.

## Acknowledgments

We thank Google for supporting Preksha Nema through their Google India Ph.D. Fellowship program. We thank our anonymous reviewer for suggesting the butterfly example which is mentioned in the introduction.

## References

[Chen *et al.*, 2016] Danqi Chen, Jason Bolton, and Christopher D. Manning. A thorough examination of the cnn/daily mail reading comprehension task. In *ACL (1)*. The Association for Computer Linguistics, 2016.

[Chung *et al.*, 2014] Junyoung Chung, Çaglar Gülçehre, KyungHyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *CoRR*, abs/1412.3555, 2014.

[Cui *et al.*, 2017] Yiming Cui, Zhipeng Chen, Si Wei, Shijin Wang, Ting Liu, and Guoping Hu. Attention-over-attention neural networks for reading comprehension. In *ACL (1)*, pages 593–602. Association for Computational Linguistics, 2017.

[Dhingra *et al.*, 2017] Bhuwan Dhingra, Hanxiao Liu, Zhilin Yang, William W. Cohen, and Ruslan Salakhutdinov. Gated-attention readers for text comprehension. In *ACL (1)*, pages 1832–1846. Association for Computational Linguistics, 2017.

[Hermann *et al.*, 2015] Karl Moritz Hermann, Tomás Kociský, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. Teaching machines to read and comprehend. In *NIPS*, pages 1693–1701, 2015.

[Hill *et al.*, 2015] Felix Hill, Antoine Bordes, Sumit Chopra, and Jason Weston. The goldilocks principle: Reading children's books with explicit memory representations. *CoRR*, abs/1511.02301, 2015.

[Hu *et al.*, 2017] Minghao Hu, Yuxing Peng, and Xipeng Qiu. Mnemonic reader for machine comprehension. *CoRR*, abs/1705.02798, 2017.

[Joshi *et al.*, 2017] Mandar Joshi, Eunsol Choi, Daniel S. Weld, and Luke Zettlemoyer. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. In *ACL (1)*, pages 1601–1611. Association for Computational Linguistics, 2017.

[Kadlec *et al.*, 2016] Rudolf Kadlec, Martin Schmid, Ondrej Bajgar, and Jan Kleindienst. Text understanding with the attention sum reader network. In *ACL (1)*. The Association for Computer Linguistics, 2016.

[Khashabi *et al.*, 2016] Daniel Khashabi, Tushar Khot, Ashish Sabharwal, Peter Clark, Oren Etzioni, and Dan Roth. Question answering via integer programming over semi-structured knowledge. In *IJCAI*, pages 1145–1152. IJCAI/AAAI Press, 2016.

[Kociský *et al.*, 2017] Tomás Kociský, Jonathan Schwarz, Phil Blunsom, Chris Dyer, Karl Moritz Hermann, Gábor Melis, and Edward Grefenstette. The narrativeqa reading comprehension challenge. *CoRR*, abs/1712.07040, 2017.

[Lai *et al.*, 2017] Guokun Lai, Qizhe Xie, Hanxiao Liu, Yiming Yang, and Eduard H. Hovy. RACE: large-scale reading comprehension dataset from examinations. In *EMNLP*, pages 785–794. Association for Computational Linguistics, 2017.

[Nguyen *et al.*, 2016] Tri Nguyen, Mir Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, and Li Deng. Ms marco: A human generated machine reading comprehension dataset. November 2016.

[Onishi *et al.*, 2016] Takeshi Onishi, Hai Wang, Mohit Bansal, Kevin Gimpel, and David A. McAllester. Who did what: A large-scale person-centered cloze dataset. In *EMNLP*, pages 2230–2235. The Association for Computational Linguistics, 2016.

[Pennington *et al.*, 2014] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. Glove: Global vectors for word representation. In *EMNLP*, pages 1532–1543. ACL, 2014.

[Rajpurkar *et al.*, 2016] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. Squad: 100, 000+ questions for machine comprehension of text. In *EMNLP*, pages 2383–2392. The Association for Computational Linguistics, 2016.

[Richardson *et al.*, 2013] Matthew Richardson, Christopher J. C. Burges, and Erin Renshaw. Mctest: A challenge dataset for the open-domain machine comprehension of text. In *EMNLP*, pages 193–203. ACL, 2013.

[Seo *et al.*, 2016] Min Joon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. Bidirectional attention flow for machine comprehension. *CoRR*, abs/1611.01603, 2016.

[Trischler *et al.*, 2016] Adam Trischler, Tong Wang, Xingdi Yuan, Justin Harris, Alessandro Sordoni, Philip Bachman, and Kaheer Suleman. Newsqa: A machine comprehension dataset. *CoRR*, abs/1611.09830, 2016.

[Wang *et al.*, 2017] Wenhui Wang, Nan Yang, Furu Wei, Baobao Chang, and Ming Zhou. Gated self-matching networks for reading comprehension and question answering. In *ACL (1)*, pages 189–198. Association for Computational Linguistics, 2017.

[Xiong *et al.*, 2016] Caiming Xiong, Victor Zhong, and Richard Socher. Dynamic coattention networks for question answering. *CoRR*, abs/1611.01604, 2016.