Multiple Choice Machine Reading Comprehension

IMT2017003, IMT2017522

May 2020

1 Abstract

The project is based on the problem statement of Machine Comprehension. We restricted ourselves to short-context single-hop reasoning machine comprehension tasks.

2 Introduction

Machine comprehension is one of the core verticals in the Natural Language Processing domain, primarily as it serves as an indicator of quantifying machine reasoning. The problem statement can be formalized as follows, there is a context (usually short in length, containing up to a few paragraphs) followed by a question based on the context. The model is supposed to be able to answer the question correctly (where the correctness is defined by the context). For most of our study we considered a multiple-choice question scenario where the model is also provided with a set of candidate answers to the question. The model here needs to select the correct answer amongst the candidate answers.

3 Contents

- GenNet : Reading Comprehension with Multiple Choice Questions using Generation and Selection model[3]
- Improving Machine Reading Comprehension via Adversarial Training[[8]

4 Study

4.1 Attention Pooling Networks

Intuitively, the problem of Reading Comprehension hinges on the contextual information present in the passage (question and answer) with respect to question (passage and answer) and answer (passage and question). Hence, we would aim towards a mechanism which can capture the joint context between two entities. This points in the direction of an attention mechanism where for example : if we have two paragraphs, p_1 and p_2, the refined representation of p_1 should be defined based on attention over p_2 and similarly for p_2 . For this purpose, we looked at Attention Pooling Networks. These have shown success in the realms of discriminative model training in terms of pairwise ranking or classification with neural networks. Attention Pooling enables the pooling layer to be aware of the current input pair in a way that information from the two input items can directly influence the computation of each other's representation. The model has also shown success in matching pairs of input with significant length variations, a feature very desirable to Reading Comprehension as the context and question/answers vary significantly in length.
We can define the mechanism as follows :
Input: Let A and B be two entities which we want to pool (for a more concrete scenario, A can be context and B can be question related to the context).
We get the word embedding for the two entities.
We pass A and B (individually) to a bi-LSTM (or bi-GRU) where the dimensions of hidden-size is denoted as H. The output from each time step is the concatenation of the two output vectors from both directions (H + H - which we shall define as c). Doing so we obtain the following :
A as a matrix where $A \in \mathbb{R}^{cxM}$
where c is the dimension of word-embedding for each token and M is the number of tokens in A. Similarly, B as a matrix where$B \in \mathbb{R}^{cxL}$. Having obtained A and B we use

$$G = tanh(A^T UB) \tag{1}$$

where $G \in \mathbb{R}^{MxL}$.

Next, we apply row-wise max-pooling (column-wise max-pooling) to obtain gB (gA). Here, each element j in the vector g_B can be interpreted as an importance score for the context around the j^{th} word in B with regard to A.
Next, we apply the softmax layer to the vectors g_A and g_B to create attention-vector f_A and f_B where $f_A \in \mathbb{R}^M$ and $f_B \in \mathbb{R}^L$

Finally, we can obtain the refined embedding for A and B by :

$$r^A = matrixmultipy(A, f_A)$$
$$r^B = matrixmultipy(B, f_B)$$

4.2 Attention-over-Attention Neural Networks

This paper talks about the Cloze-style reading comprehension problem. In this variant of the machine reading comprehension problem, the answer to the query is one word which is already present in the document. More formally, given a tuple $(\mathcal{D}, \mathcal{Q}, \mathcal{A})$, $\mathcal{A} \in \mathcal{D}$ i.e, a document, a query and the answer. The authors of this paper were the first ones to present an "attention-over-attention" mechanism which outperformed state of the art results by a large margin when this paper was released.

4.2.1 Contextual Embeddings

The first step is to transform the document, $\mathcal{D}$ and the query $\mathcal{Q}$ into one-hot representations which are then converted to continuous representations into a shared embedding matrix W. Then 2 bi-directional RNNs (GRUs) are used to capture the backward and forward contextual embeddings which are then concantenated together.

4.2.2 Pair-Wise Matching Score

We create a matrix $M(i, j)$ which is $h_{doc}(i)^T * h_{query}(j)$. This matrix has the dimensions $|\mathbb{R}^{\mathcal{D}*\mathcal{Q}}|$

4.2.3 Attention

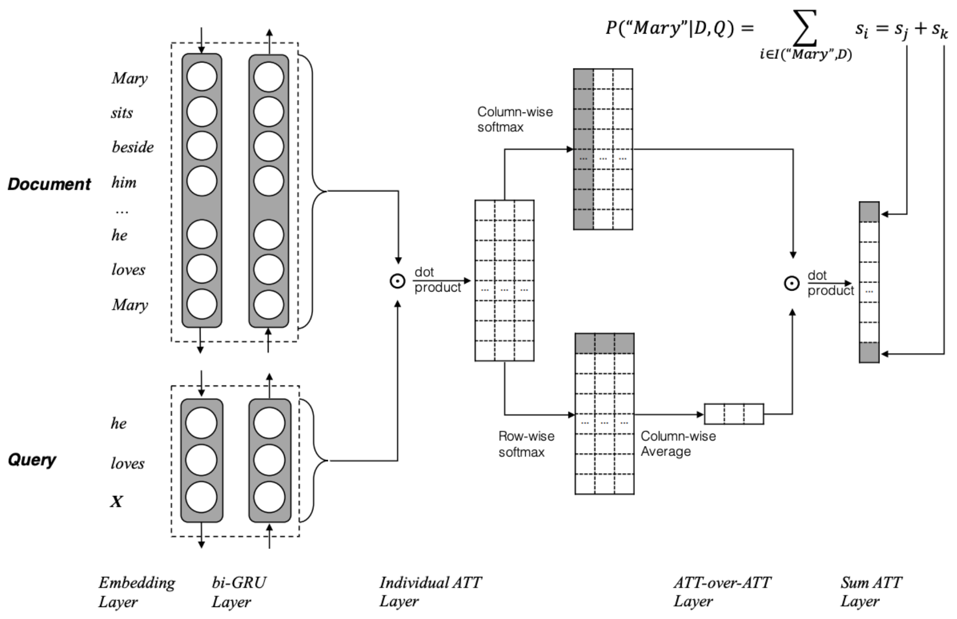
We then apply a column-wise softmax over the matrix M to get the $query - to - document$ attention, denoted as $\alpha(t) \in \mathbb{R}^{|\mathcal{D}|}$

4.2.4 Attention-over-Attention

We now apply a row-wise softmax over the matrix M to get the $document - to - query$ attention denoted as $\beta(t) \in \mathbb{R}^{|\mathcal{Q}|}$. Now, we take the average over all t for each $\beta(t)$ and denote it as β.

We then get $s \in \mathbb{R}^{\mathcal{D}} = \alpha^T \beta$

Intuitively, this operation is calculating a weighted sum of each individual document-level attention $\alpha(t)$ when looking at query word at time t. This way, the contributions by each query word can be learned explicitly, and the final decision (document-level attention) is made through the voted result by the importance of each query word.



4.2.5 Final Prediction

Finally, we use the $sum - attention$ mechanism to get the final result.

$$P(w|\mathcal{D}, \mathcal{Q}) = \sum_{i\in I(w,\mathcal{D})} s_i, w \in V \tag{2}$$

where $I(w, D)$ indicate the positions that word w appears in the document $\mathcal{D}$. The log-likelihood is then maximised.

The candidate word with the highest probability can be picked as the answer. Additional checks can be conducted to see if the word chosen is suitable grammatically. If the word isn't, the second best word can be chosen and so on. Another alternative is to use the N-best ranking strategy.

4.2.6 N-best re-ranking strategy

1. Make an N-best list for alternative candidates
2. Refill candidates into the query and score each of the sentences
3. The sentence scoring can be done in 3 ways
   (a) Global N-Gram LM - Trained on the document training data, it is used to score the fluency of the sentence.

   (b) Local N-Gram LM - Trained sample by sample on the test data, as it aims to explore the information in the given document. It is useful when there are a lot of OOV(Out of Vocabulary) words present.

   (c) Word-Class LM - Similar to Global N-Gram LM, it is also trained on the document part of the training data.

4. A weighted sum of the above features is calculated and the candidate with the lowest cost is chosen as the answer.

4.3 DCMN

DCMN - Dual Co-Matching Network for Multi-Choice Reading Comprehension. Since the task of reading comprehension hinges on obtaining the join context, intuitively for a multi-choice question setting we need to model a three-way joint-contexts per answer candidate, these would include :

1. Passage-Question context
2. Question-Candidate answer context
3. Candidate answer-Passage context

This needs to be done for all the candidate options. DCMN models the relationship among passage, question and answer bidirectionally.
The first layer is an encoder which encodes each token in passage (question, and answer) into a fixed length vector following a language-model (the origin paper has used BERT for this task). Hence, for each candidate answer, we get something along these lines :

$$\begin{aligned}
H^p &= Bert(P) \in \mathbb{R}^{pxl} \\
H^q &= Bert(Q) \in \mathbb{R}^{qxl} \\
H^a &= Bert(A) \in \mathbb{R}^{axl}
\end{aligned}$$

where we have denoted the passage, question and answer embedding respectively (l is dimension of word embedding and p,q and a are the dimensions of passage, question and answer sequences - fixed for each type)
A bidirectional matching strategy and a gating mechanism has been used to get all pair-wise matching representation among the triplet (p and q, p and a, p and q). The concept of matching is similar to attention-pooling networks where it is applied to all combinations of the triplet.

$$\begin{aligned}
G^{pa} &= softmax(H^p W (H^a)^T) \\
E^p &= G^{pa} H^a \\
E^a &= (G^{pa})^T H^p \\
S^p &= ReLU(E^p W1) \\
S^a &= ReLU(E^a W2)
\end{aligned}$$

where W, W_1 and W_2 are all learnable parameters with dimensions R^{lxl}
$G^{qa} \in \mathbb{R}^{pxa}$ represents the attention weight matrix between the passage and the answer and $E^p \in \mathbb{R}^{pxl}$ *and* $E^a \in \mathbb{R}^{axl}$ represent answer-aware passage representation and passage-aware answer representation respectively.
To combine the two representations and obtain a single passage-answer representation we use a gating mechanism (instead of simple concatenation).

$$
\begin{aligned}
&M^p = MaxPooling(S^p)\\
&M^a = MaxPooling(S^a)\\
&g = SoftMax(M^pW_3 + M^aW_4 + b)\\
&M^{pa} = g * M^p + (1-g) * M^a
\end{aligned}
$$

Here, M^{pa} denotes the final bidirectional matching representation for the passage answer sequence pair.
Similarly, we obtain the passage-question vector and question-answer vector. Now, we have 3 vectors of dimensions l. We concatenate the three vectors to obtain one vector which captures the triplet of relations. This is used as our feature vector and passed into an ANN for answer prediction.
Hence, on a broad level our problem resolves to a binary classification problem each data-point with n candidate options being dealt as n data-points. The output is either a 0 (for incorrect options) and 1 for correct options.
This suffers from two flaws :

1. It considers each Question-Passage-Candidate triplet as a separate data-point where the relations between candidate-options are lost. Intuitively, this would be a plus-point while dealing with situations such as multiple-correct options.

2. The data-distribution for single-correct data-set gets skewed with 75% negative data-points and 25% positive points.

4.4 DUMA

DUMA : DUal Multi-head co-Attention for multiple choice reading comprehension. This is based on a simplified approach to capture the information from the triple of passage-question-candidate answer for multiple choice reading-comprehension.
The literature advocates a two-phased approach to MCRC (Multiple-Choice Reading Comprehension) -

1. Representation Encoding.

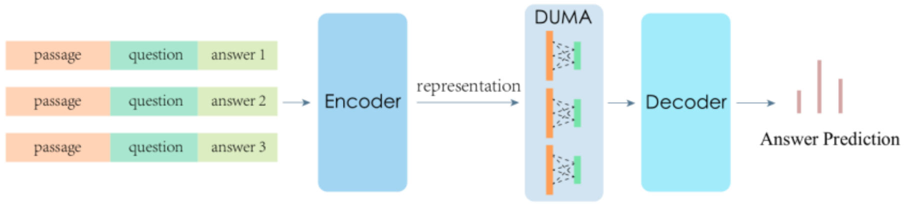
2. Capturing the information between the triplet of question-passage-answer.

Quite intuitively attention-mechanism is the go-to methodology for both the tasks. They have been used heavily in capturing inter-contextual information.

Transformers use self-attentions mechanism to represent dependencies and relationships of different positions in one single sequence which is an efficient method to obtain representations of sentences for global encoding. However, it does not capture inter-dependencies between entities, something crucial for MCRC.
For the present scenario, the problem statement is formalized as follows :

1. A passage: P
2. A question: Q
3. A set of candidate answers: $< A_1, \ldots, A_t >$

And our aim is for the model to learn the following : $Prob(A_1, \ldots, A_t|P, Q)$
The workflow is as follows :



For each data-point, the model creates t-sequences (one for each candidate answer) by concatenating the passage, question with each candidate answer. The sequence is passed into an encoder which (which is a pre-trained language model).

$$P = \{p_1, \ldots, p_m\}$$

where the passage is represented by m tokens

$$Q = \{q_1, \ldots, q_n\}$$
$$A_i = \{a_1, \ldots, a_k\}$$

The input to the encoder is - $P \bigoplus Q \bigoplus Ai$
The output (E) represents the globalized contextual representation for each candidate answer, and it belongs to $\mathbb{R}^{l(m+n+k)}$ where l denotes the factor brought in by the language model
We separate E into two parts -

1. $E^p = [e1^p, ..., (e(lp))^p]$
2. $E^{qa} = [e1^{qa}, ..., (e(lqa))^{qa}]$

where E^p denotes the representation for the passage and E^{qa} denotes the representation of the question-answer.
We then place attention heads to calculate the attention representation in a bidirectional manner i.e., one with E^p as the Query and E^{qa} as the Key and Value and another with E^{qa} as the Query and E^p as the Key and Value.

Multiple such heads are used for each direction and the output is concatenated and projected into a lower-dimensional space using a learn-able parameter matrix.

$$head_i = Attention(E^p {W_i}^q, E^{qa} {W_i}^k, E^{qa} {W_i}^v)$$
$$Multi - HeadedAttention(MHA) = Concat(head_1, head_2, ...head_h)W^o$$
$$MHA_1 = MHA(E^p, E^{qa}, E^{qa})$$
$$MHA_2 = MHA(E^{qa}, E^p, E^p)$$
$$DUMA(E^p, E^{qa} = Fuse(MHA_1, MHA_2)$$

The Fuse function first obtains the pooled sequence using mean pooling and the fuses the two entities using element wise simple operations.
The output vector is used as the feature vector corresponding to each of the passage-question-candidate answer triplet.
The decoder uses this vector to obtain the output using an ANN. The problem is resolved into a binary classification problem where each data-points are dealt as t data-points (t representing the number of candidate answers).

4.5 Match-LSTM

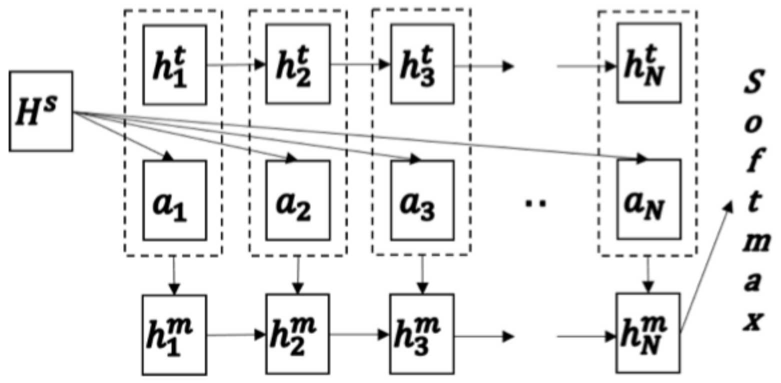
Match-LSTM has been used to solve the Natural Language Interface (NLI) problem. Given 2 sentences, P and H, the NLI problem is to determine whether the premise sentence P can determine the hypothesis sentence H.

Past works include embedding sentences using an LSTM and then adding a multi layer neural network or a neural attention model on top of these embeddings to get the result. A fundamental flaw in the above approach is that not all words in the sentence are important. To contradict the hypothesis, a single mismatch would be enough.

The Match-LSTM model takes a different approach. The LSTM tries to match each word of the hypothesis H with the attention weighted representation of the premise P. Matched results can be 'remembered' or 'forgotten'. Say we have a premise "$A\ boy\ likes\ food$" and a hypothesis "$A\ girl\ likes\ eating.$". Now, if we look at the words "*boy*" and "*girl*", it's enough to show that's it's a mismatch. This mismatch will have to be scored highly. On the other hand, matches like "*A*" or "." don't really matter as much. They'll have to be scored low. This is where attention comes in.

Match-LSTM has 2 improvements over the previous state of the art model by Rocktäschel et al. (2016)

1. Rocktäschel's model used only the single vector representation for matching the premise and the hypothesis. Instead, mLSTM uses the attention



weighted vectors instead to match against the hypothesis. This is done by using an RNN to determine how well the sentences are matched until a certain position. In the end the RNN will produce a single vector representing the matching of the two entire sentences.

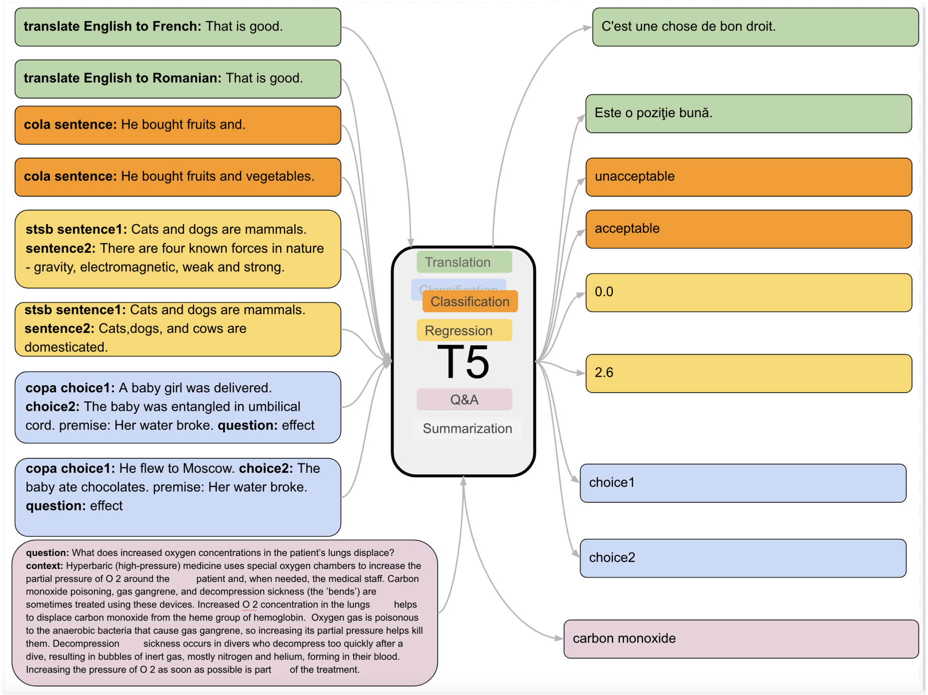
2. Rocktäschel's model did not allow emphasis to be placed or removed as shown in the example earlier. To accomodate for this, apart from using the hidden states of the premise h_j^s , and the hypothesis h_k^t, to calculate the matching between the j^{th} word of the premise and the k^{th} word of the hypothesis, a new hidden state h_{k-1}^m is introduced. This helps in "remembering" the important results and "forgetting" the non-essential matches. The concatenation of a_k, which is the attention-weighted version of the premise for the k^{th} word of the hypothesis, and h_k^t the hidden state of the k^{th} word itself is used as an input to the mLSTM.

Finally, the last hidden state, h_N^m is used to predict whether the word is a match or a non-match.

4.6 UnifiedQA

UnifiedQA is an attempt to bring together different question-answer tasks under one umbrella. The Reading Comprehension tasks have seen various subdivisions such as, Multiple Choice Reading Comprehension (where a question is followed my multiple candidate-answers), Extractive Reading Comprehension (where the answer corresponds to a particular piece of text from the context), Yes/No Reading Comprehension (where the answer is either a yes or a no), etc. This has lead to work being done in these niche divisions. This also hinders the ability to exploit multiple data-sets spanning different formats. The paper argues that these divisions are superficial basing on the intuition while question format and relevant knowledge may vary across QA data-sets, the underlying linguistic understanding and reasoning abilities are largely common. Here, they

introduce the UnifiedQA model - a single pre-trained question-answer model which performs well across 17 data-sets spanning 4 diverse formats. It also performs well on unseen data-sets showing strong generalization. Fine tuning the model to specified tasks results in a new state-of-the-art on 6 data-sets and hence, this proves as a strong starting point for building QA systems.
This is largely inspired from Google's Text-to-text-Transfer-Transformer(T5) which is an encoder-decoder transformer architecture which aims to bring a variety of natural-language processing tasks under the same umbrella.


translate English to French: That is good.

translate English to Romanian: That is good.

cola sentence: He bought fruits and.

cola sentence: He bought fruits and vegetables.

stsb sentence1: Cats and dogs are mammals. **sentence2:** There are four known forces in nature - gravity, electromagnetic, weak and strong.

stsb sentence1: Cats and dogs are mammals. **sentence2:** Cats,dogs, and cows are domesticated.

copa choice1: A baby girl was delivered. **choice2:** The baby was entangled in umbilical cord. premise: Her water broke. **question:** effect

copa choice1: He flew to Moscow. **choice2:** The baby ate chocolates. premise: Her water broke. **question:** effect

question: What does increased oxygen concentrations in the patient's lungs displace? **context:** Hyperbaric (high-pressure) medicine uses special oxygen chambers to increase the partial pressure of O 2 around the patient and, when needed, the medical staff. Carbon monoxide poisoning, gas gangrene, and decompression sickness (the 'bends') are sometimes treated using these devices. Increased O 2 concentration in the lungs helps to displace carbon monoxide from the heme group of hemoglobin. Oxygen gas is poisonous to the anaerobic bacteria that cause gas gangrene, so increasing its partial pressure helps kill them. Decompression sickness occurs in divers who decompress too quickly after a dive, resulting in bubbles of inert gas, mostly nitrogen and helium, forming in their blood. Increasing the pressure of O 2 as soon as possible is part of the treatment.

The T5 transformer takes the format of the input as an input itself, the intuition behind it being that the knowledge of determining the format falls under natural-language processing as well. The transformer then determines the format and models its behavior accordingly. This ability allows UnifiedQA to exploit a large amount of data as it can choose its data-sets to include data cutting across various formats.
Suppose we have K question-answering formats $(F_1, \ldots, F_k)$, and for each format have K_i data-sets, i.e. $({D_{k1}}^1, \ldots, {D_{kk}}^k)$
Each data-set is divided into training examples and evaluation examples:

$$({T_j}^i, {E_j}^i) = {D_{kj}}^i$$

The text-to-text paradigm is used to convert each training question q in format F_i into a plain text-representation
$-\quad$ each q from $F_i \longrightarrow enc_i(q)$

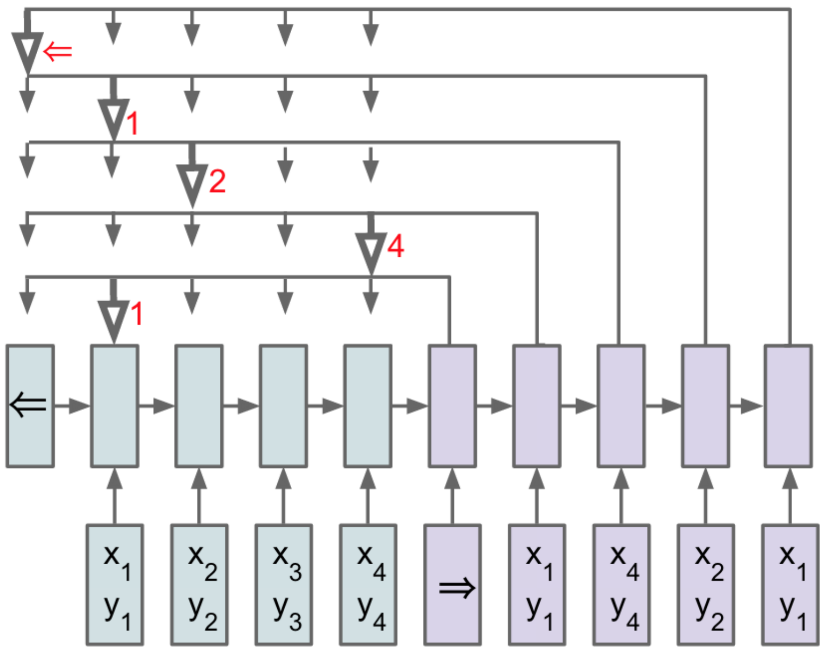
Doing this we obtain a mixed-common training pool from all the datasets -

$$T = \bigcup_{i=1}^{k} \bigcup_{j=1}^{k_i} (enc_i(q) \mid q \ \in \ {T_j}^i) \tag{3}$$

The training batches are drawn from here while considering the size of individual data-sets, such that the probability of a data-point appearing is uniformly distributed among all the sets.
Each target dataset is converted to a text-in
text-out format. The question always comes first, followed by the context
options or both. Each part is separated by a line-break ('
n') to not make the format overly specific. While it is inspired from the T5 model, a key difference here is that it does not provide the question-answer format explicitly through any extra prefix-tokens, instead it expects the model to learn to infer the format from the data-itself. The metrics for the performance are defined with respect to different formats. The success of UnifiedQA suggests that it can be used as a starting-point for various Reading Comprehension tasks.

4.7 Pointer Networks

The existing neural-network architectures were designed for obtaining output over a pre-defined output dictionary. However, in various tasks, for example in extractive question-answer tasks, the answer is an arbitrary length sequence present somewhere in the passage, the output dictionary is dynamic (for our example, different for different passages). Pointer-Network uses the existing attention framework to learn the conditional probability of an output sequence with elements that are discrete tokens corresponding to positions in the input sequence. It utilizes attention as a pointer to select a member of the input sequence as the output, and it does so effectively while maintaining the simplicity of fitting right in the concept of existing attention mechanism.



The traditional sequence-to-sequence models formulate the problem as follows: Suppose we have an input denoted by P and the output dictionary has a dimension of M, then the probability modelled by the network is :

$$Prob(C_i|C_1, C_2, \ldots, C_{i-1}, P)$$

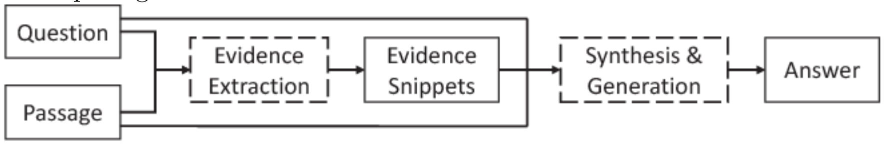
where all C_i belong to the set of pre-defined tokens.
To solve this problem, pointer networks models this probability as follows :

$$\begin{aligned} u_j{}^i &= v^T(tanh(W_1 e_j + W_2 d_i)) \; where \; j \in \{1, \ldots, n\} \\ Prob(C_i|C_1, C_2, \ldots, C_{i-1}, P) &= softmax(u^i) \end{aligned}$$

where v W_1 *and* W_2 are all parameters which can be learned.
This readily provides a way to tackle distribution over dynamic-input determined dictionaries. For our problem statement of Reading Comprehension, this has been used for extractive question-answering as well as feature extraction for other formats.

4.8 S-Net

S-Net primarily targets Reading Comprehension tasks where the context can span up-to multiple passages and the answer is not necessarily present (in terms of exact terms) in the passage. It also does not consider the presence of candidate options, so it targets an answer-generation task rather than an answer-classification task. The paper proposes a two-phased framework which involves extraction, followed by synthesis of answer using the extraction. The model first identifies most important sub-spans from the passage as evidence, and the answer synthesis model takes, the evidence as additional features along with the question and passage to further elaborate the final answers. The answer extraction model uses a state of the art neural network for single passage reading comprehension and proposes a dual-learning framework involving sub-span extraction along-side an additional task of passage ranking to bolster it for the multi-passage scenario.



The working pipeline is as shown,

1. The Question and Passage are taken as input
2. Evidence is extracted from the passage
3. Evidence snippets are obtained using pointer-networks
4. The question and the passage along with the output from the pointer-net is used for generating answer

4.8.1 Evidence Extraction

The multi-task learning framework comes in here. As the passage can span to multiple paragraphs, passage ranking is introduced prior to text-span prediction. The initial token representation is obtained for each of paragraph and question by utilizing a character-level approach and a word-level approach.

1. ${u_t}^Q = BiGRU({u_{t-1}}^Q, [{e_t}^Q, {char_t}^Q)$ - Question Representation
2. ${u_t}^P = BiGRU({u_{t-1}}^P, [{e_t}^P, {char_t}^P)$ - Passage Representation

where, e_t corresponds to word embedding and $char_t$ corresponds to character embedding for the word (which has also been passed through a Bidirectional-GRU).
Next, having the passage and question representation, we obtain a sentence-pair representation using soft-alignment of the words in the two, this is denoted using v^P

$${v_t}^P = GRU({v_{t-1}}^P, {c_t}^Q)$$

where ${c_t}^Q$ is an attention-pooling vector of the whole question u^Q

$${c_t}^Q \ = \ att(u^Q, [{u_t}^P, {v_{t-1}}^P])$$

Then, we use the match-LSTM to add gating to the input denoted by $[u_t{}^P, c_t{}^Q]$ to determine the importance of passage parts :

$$g_t = sigmoid(W_g[u_t{}^P, c_t{}^Q])$$
$$[u_t{}^P, c_t{}^Q]^* = g_t{}^{\cdot}[u_t{}^P, c_t{}^Q]$$

and finally, the passage representation is obtained as :

$$v_t{}^P = GRU(v_{t-1}{}^P, [u_t{}^P, c_t{}^Q]^*$$

Having obtained this passage-representation, Pointer-Networks are used to predict the position of evidence snippets. The passages are concatenated together in order to predict one-span for the evidence snippet.
The complete concatenated passage representation is denoted by: $v_t{}^P$ where $t \in \{1, \ldots, N\}$ where N being the sum of length of all the passages. The attention mechanism is utilized as a pointer to predict the start-position and end-position of the evidence snippet.

$$s_j{}^t = v^T(tanh(W_h{}^P v_j{}^P + W_h{}^a h_{t-1}{}^a)$$
$$a_i{}^t = exp(s_i{}^t)/NormalizationFactor$$
$$p^t = argmax(a_1{}^t, ..., a_N{}^t)$$

The input of the answer recurrent network is the attention-pooling vector based on current predicted probability a^t

$$c_t = Sum(i- > 1\ to\ N)a_i{}^t v_i{}^P$$
$$h_t{}^a = GRU(h_{t-1}{}^a, c_t)$$

For the answer recurrent network, the first initial state is determined using r^Q, which is attention pooling vector of the question based on the parameter $v_r{}^Q$
The loss function is determined using whether the network predicts the correct span-positions or not.

4.8.2 Passage Ranking

Using r^Q we apply attention-pooling to obtain a representation of the passage - r^P. Then we combine both these representations of the question and the passage and pass them through 2 fully connected layers to obtain a matching score. The problem is essentially dealt as a binary classification.

4.8.3 Joint Learning

A convex combination of the loss metrics from evidence extraction and passage ranking is obtained to determine the joint-loss metric.

4.8.4 Answer Synthesis

4.9 Gen-Net

Gen-Net builds on top of the S-Net ideology and its model. It tackled Multi-Choice Reading Comprehension format specifically. Intuitively it defines two approaches towards tackling the problem:

1. Generating an answer
2. (a) Obtaining the best match
   (b) Eliminating the worst match

It does the following under the intuition that human-processing of Multi-Choice Questions follow along the same lines. The proposed model mimics the answer generation and then matching process.
The paper proposes the use of S-Net as the answer generation model as it not only finds the answer from the passage, but it can also synthesize passage when required. Some questions are tricky and there answer lies in different span of passage. In such situations S-net has been shown to be effective.
We generate an answer through the S-Net model using the passage and question, the generated answer is denoted as $A_n = [a_1, a_2, ...a_n]$
The scoring mechanism works as follows, for each option, the bi-linear similarity with respect to the generated answer is obtained.

$$score(i) = a_t W_{att} z_{t_i}$$

where i is the number of option, a_t is the generated answer vector, z_{t_i} is the option vector and W_{att} is a learnable parameter. The option corresponding to the largest score is selected and the cross-entropy loss is used as the loss-metric.

4.10 Adversarial Training

A number of neural network models display bad robustness against perturbations. Such perturbations which confuse a network are termed as Adversarial examples. To counter this, adversarial training was proposed to make the models more robust, and to incorporate for the same, including adversarial examples during training was suggested. Adversarial training (AT) as a regularization task has proved its effectiveness across various task formats, such as image classification and text classification, etc. Such methods have shown to bring a significant and universal improvement in performance. Though it has been shown ineffective in defending the model against artificial adversarial examples, it helps boost the model's performance. The aim of this study is to generalize adversarial training to NLP problems which could lead to a performance boost across task domains.

The task has been defined as follows:
Three formats of Reading Comprehensions tasks were considered :

1. Span-Extraction Reading Comprehension
2. Span-Extraction Unanswerable Reading Comprehension (same as the first except that the question might even be unanswerable)
3. Multiple-Choice Reading Comprehension

As a starting point we consider tokenized representation for the Question and Passage (these are common across all three tasks) : $Q = [q_1, ..., q_n]$ and $P = [p_1, ..., p_m]$
For the current study, a fine-tuned BERT model has been used as the base model with appropriate adaptation for the specific task format.
For the formats consisting of passage and questions, the passage and question are concatenated separated by a $[SEP]$ token. For the MCRC format, we obtain one-data point with respect to each candidate option where the passage followed by question followed by the candidate option is concatenated (using $[SEP]$ token here as well)
Let us take the MCRC task in particular, suppose we have a passage, question and m corresponding candidate options. Then we obtain :

$$X^i = [CLS] < P > [SEP] < Q > [SEP] < O^i > i \in [m]$$

This is fed as the input to BERT and the output is obtained from the Bert pooler :

$$H^{'} = [B^1, B^2, ...B^m] \text{ where } H^{'} \in R^{mxh}$$

The final prediction is then obtained by applying linear transformation followed by softmax over the m options of $H^{'}$ and the loss metric used is cross-entropy.

4.10.1 Introducing Adversarial Training

Adversarial examples has shown to not only improve the robustness of the classifier in various different domains, but it also boosts the network's performance on clean inputs. For AT, we first construct adversarial examples by generating worst-case perturbations that maximize the current loss function, and then train the model on both of clean examples and adversarial examples.
For the particular case of MCRC, the inputs are a sequence of words/tokens and we define perturbations at this level of word/token embedding. Suppose we have an input sequence which we denote by x, then:

$$x = [x_o, ..., x_{l-1}] \in R^{lxh}$$

where l is the number of tokens in the sequence and h is the dimension of the word embedding.

Let y denote the target, then the worst case perturbation, denoted by r_{AT} is the one that maximizes the loss with a bounded norm :

$$r_{AT} = argmax_{||r||<\epsilon}(L(x + r; y; \theta))$$

where x denotes the input, r denotes the perturbation, y denotes the target and θ denotes the trainable parameters. The exact value of r_{AT} is untractable, hence we use the following instead:

$$r_{AT} = \epsilon(g||g||) \text{ where } g = \delta x L(x; y; \theta^{'})$$

The adversarial example is constructed as :
$x_{AT} = x + r_{AT}$ where ϵ is the hyperparameter which controls the relative strength of the perturbations.
On each batch of inputs, the model is trained on both clean examples and adversarial examples by minimizing the loss on both the examples simultaneously. This has shown to boost the performance of the model, especially on how it deals with low-frequency words. AT and Data-augmentation has also been shown to be orthogonal to each other and hence, the two can be utilized together.

References

[1] Yiming Cui, Zhipeng Chen, Si Wei, Shijin Wang, Ting Liu, and Guoping Hu. Attention-over-attention neural networks for reading comprehension. *CoRR*, abs/1607.04423, 2016.

[2] Cícero Nogueira dos Santos, Ming Tan, Bing Xiang, and Bowen Zhou. Attentive pooling networks. *CoRR*, abs/1602.03609, 2016.

[3] Vaishali Ingale and Pushpender Singh. Gennet : Reading comprehension with multiple choice questions using generation and selection model, 2020.

[4] Daniel Khashabi, Tushar Khot, Ashish Sabharwal, Oyvind Tafjord, Peter Clark, and Hannaneh Hajishirzi. Unifiedqa: Crossing format boundaries with a single qa system, 2020.

[5] Chuanqi Tan, Furu Wei, Nan Yang, Bowen Du, Weifeng Lv, and Ming Zhou. S-net: From answer extraction to answer generation for machine reading comprehension, 2017.

[6] Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. Pointer networks, 2015.

[7] Shuohang Wang and Jing Jiang. Machine comprehension using match-lstm and answer pointer, 2016.

[8] Ziqing Yang, Yiming Cui, Wanxiang Che, Ting Liu, Shijin Wang, and Guoping Hu. Improving machine reading comprehension via adversarial training, 2019.

[9] Shuailiang Zhang, Hai Zhao, Yuwei Wu, Zhuosheng Zhang, Xi Zhou, and Xiang Zhou. Dcmn+: Dual co-matching network for multi-choice reading comprehension, 2019.

[10] Pengfei Zhu, Hai Zhao, and Xiaoguang Li. Dual multi-head co-attention for multi-choice reading comprehension, 2020.