Machine Learning for Machine Reading Comprehension

- Bhavya Patwa (1401063)
  - BTP under Dr. Ganesh Ramakrishnan
  - Mentor: Pankaj Singh

AHMEDABAD UNIVERSITY

Global Education at Local Cost, Context and Ethos™

Index

1. Introduction to Question Answering Systems
2. Datasets
3. Preprocessing
4. State-of-the-art research papers implemented (R-NET and S-NET)
5. Experiment
6. Results

1. Introduction to Question Answering

- Within the fields of Information Retrieval and Natural Language Processing
- Tremendous applications in e-commerce, business, search engines, customer service chatbots, visual surveillance (VQA) etc.

- Our focus: Passage Question Answering i.e. answering a question from a given set of input passages.

2. Datasets

- Stanford Question Answering Dataset (SQuAD) (Rajpurkar et al., 2016)[1]
  - 23K paragraphs from 536 Wikipedia articles
  - Q: 108K human generated, based on the paragraphs
  - A: Spans (i.e. answer is a part of the passage and start / end positions are given)
- Microsoft - Machine Reading Comprehension (MS-MARCO) (Nguyen et al., 2016)[2]
  - 1M passages from 200K+ documents retrieved using the queries
  - Q: 82k search queries (Bing)
  - A: human generated, based on the passages

3. Preprocessing

- SQuAD
  - Used GloVe[3] (840B tokens, 300d) embeddings for word representation
  - 8d character level embeddings
  - Data = {passage, question, answer, start index, end index}
  - Fixed number of passage per example: 1
  - No filtering of any examples

3. Preprocessing...

- MS-MARCO
  - Used GloVe[3] (840B tokens, 300d) embeddings for word representation
  - Data = {arbitrary no. of passages, question, answer}
  - Answer tokens may or maynot be part of the passage
  - Therefore, start and end index answer span is generated using the Longest Common Subsequence Algorithm (DP) and only those examples are considered whose ROUGE-L[4] score > 0.7. An advantage of using LCS is that it does not require consecutive matches but in-sequence matches that reflect sentence level word order.
  - 70k filtered examples out of 82k examples

ROUGE-L-F1[4] = 2*Precision*Recall/(Precision + Recall)

4. Implementation

1. R-NET[5]:
    a. Evidence Extraction (Evidence Snippet Prediction)
2. S-NET[8]:
    a. Evidence Extraction (Evidence Snippet Prediction + Passage Ranking)
    b. Answer Synthesis

4.1 Evidence Extraction: R-NET[5] and S-NET[8]

- Question-Passage Encoding:

3.1 Question and Passage Encoder

Consider a question Q = $\{w_t^Q\}_{t=1}^m$ and a passage P = $\{w_t^P\}_{t=1}^n$. We first convert the words to their respective word-level embeddings ($\{e_t^Q\}_{t=1}^m$ and $\{e_t^P\}_{t=1}^n$) and character-level embeddings ($\{c_t^Q\}_{t=1}^m$ and $\{c_t^P\}_{t=1}^n$). The character-level embeddings are generated by taking the final hidden states of a bi-directional recurrent neural network (RNN) applied to embeddings of characters in the token. Such character-level embeddings have been shown to be helpful to deal with out-of-vocab (OOV) tokens. We then use a bi-directional RNN to produce new representation $u_1^Q, \ldots, u_m^Q$ and $u_1^P, \ldots, u_n^P$ of all words in the question and passage respectively:

$$u_t^Q = \text{BiRNN}_Q(u_{t-1}^Q, [e_t^Q, c_t^Q]) \quad (1)$$

$$u_t^P = \text{BiRNN}_P(u_{t-1}^P, [e_t^P, c_t^P]) \quad (2)$$

We choose to use Gated Recurrent Unit (GRU) (Cho et al., 2014) in our experiment since it performs similarly to LSTM (Hochreiter & Schmidhuber, 1997) but is computationally cheaper.

4.1 Evidence Extraction: R-NET[5] and S-NET[8]...

- Gated GRU with attention as decoder

3.2 Gated Attention-based Recurrent Networks

We propose a gated attention-based recurrent network to incorporate question information into passage representation. It is a variant of attention-based recurrent networks, with an additional gate to determine the importance of information in the passage regarding a question. Given question and passage representation $\{u_t^Q\}_{t=1}^m$ and $\{u_t^P\}_{t=1}^n$, Rocktäschel et al. (2015) propose generating sentence-pair representation $\{v_t^P\}_{t=1}^n$ via soft-alignment of words in the question and passage as follows:

$$v_t^P = \text{RNN}(v_{t-1}^P, c_t) \tag{3}$$

where $c_t = att(u^Q, [u_t^P, v_{t-1}^P])$ is an attention-pooling vector of the whole question (u^Q):

$$s_j^t = \text{v}^{\text{T}} \tanh(W_u^Q u_j^Q + W_u^P u_t^P + W_v^P v_{t-1}^P)$$

$$a_i^t = \exp(s_i^t) / \Sigma_{j=1}^m \exp(s_j^t)$$

$$c_t = \Sigma_{i=1}^m a_i^t u_i^Q \tag{4}$$

4.1 Evidence Extraction: R-NET[5] and S-NET[8]...

Wang & Jiang (2016a) introduce match-LSTM, which takes u_t^P as an additional input into the recurrent network:

$$v_t^P = \text{RNN}(v_{t-1}^P, [u_t^P, c_t]) \tag{5}$$

To determine the importance of passage parts and attend to the ones relevant to the question, we add another gate to the input ($[u_t^P, c_t]$) of RNN:

$$g_t = \text{sigmoid}(W_g[u_t^P, c_t])$$
$$[u_t^P, c_t]^* = g_t \odot [u_t^P, c_t] \tag{6}$$

Different from the gates in LSTM or GRU, the additional gate is based on the current passage word and its attention-pooling vector of the question, which focuses on the relation between the question and current passage word. The gate effectively model the phenomenon that only parts of the passage are relevant to the question in reading comprehension and question answering. $[u_t^P, c_t]^*$ is utilized in subsequent calculations instead of $[u_t^P, c_t]$. We call this gated attention-based recurrent networks.

4.1 Evidence Extraction: R-NET[5] and S-NET[8]...

- Overview of attention mechanism

Rocktaschel et. al (2015)[6]

$$\mathbf{M}_t = \tanh(\mathbf{W}^y\mathbf{Y} + (\mathbf{W}^h\mathbf{h}_t + \mathbf{W}^r\mathbf{r}_{t-1}) \otimes \mathbf{e}_L) \qquad \mathbf{M}_t \in \mathbb{R}^{k \times L} \qquad (11)$$

$$\alpha_t = \text{softmax}(\mathbf{w}^T\mathbf{M}_t) \qquad \alpha_t \in \mathbb{R}^L \qquad (12)$$

$$\mathbf{r}_t = \mathbf{Y}\alpha_t^T + \tanh(\mathbf{W}^t\mathbf{r}_{t-1}) \qquad \mathbf{r}_t \in \mathbb{R}^k \qquad (13)$$

Note that $\mathbf{r}_t$ is dependent on the previous attention representation $\mathbf{r}_{t-1}$ to inform the model about what was attended over in the previous step (see Eq. 11 and 13).

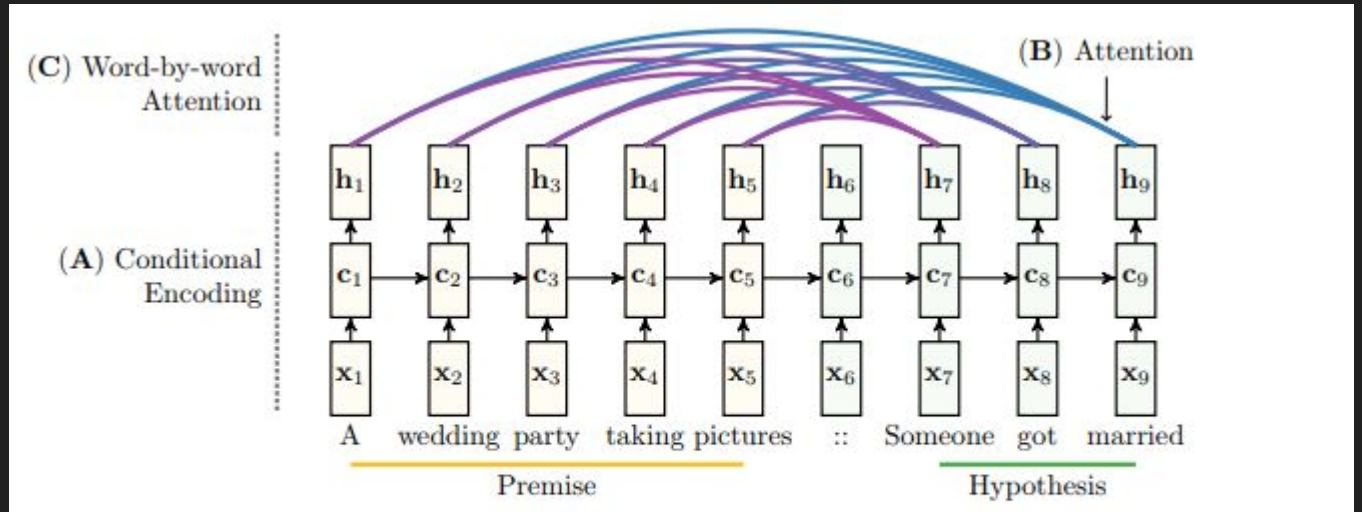
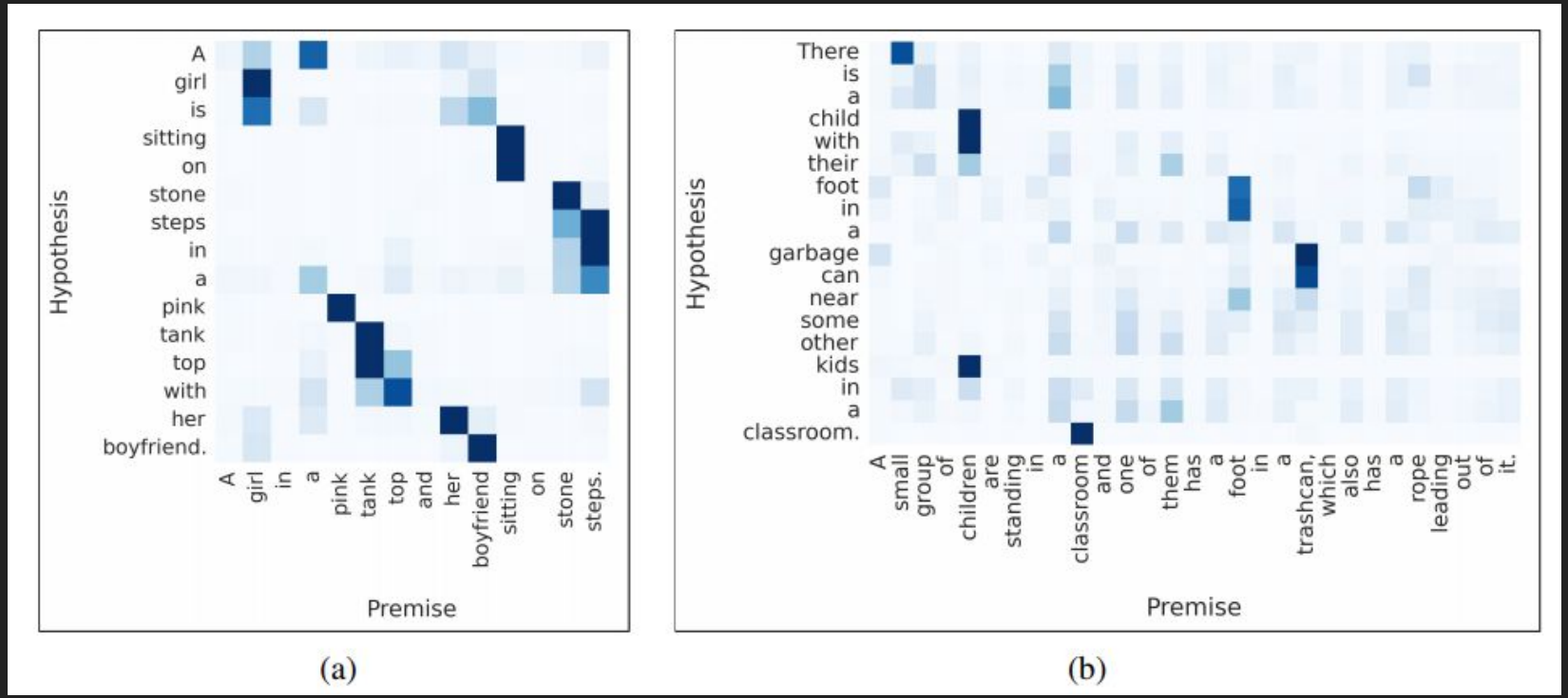


Figure 1: Recognizing textual entailment using (**A**) conditional encoding via two LSTMs, one over the premise and one over the hypothesis conditioned on the representation of the premise ($\mathbf{c}_5$), (**B**) attention only based on the last output vector ($\mathbf{h}_9$) or (**C**) word-by-word attention based on all output vectors of the hypothesis ($\mathbf{h}_7$, $\mathbf{h}_8$ and $\mathbf{h}_9$).

4.1 Evidence Extraction: R-NET[5] and S-NET[8]...

- Overview of attention mechanism

Rocktaschel et. al (2015)[6]



4.1 Evidence Extraction: R-NET[5] and S-NET[8]...

- Self-Matching attention: Not present in S-NET

3.3 SELF-MATCHING ATTENTION

Through gated attention-based recurrent networks, question-aware passage representation $\{v_t^P\}_{t=1}^n$ is generated to pinpoint important parts in the passage. One problem with such representation is that it has very limited knowledge of context. One answer candidate is often oblivious to important cues in the passage outside its surrounding window. Moreover, there exists some sort of lexical or syntactic divergence between the question and passage in the majority of SQuAD dataset (Rajpurkar et al., 2016). Passage context is necessary to infer the answer. To address this problem, we propose directly matching the question-aware passage representation against itself. It dynamically collects evidence from the whole passage for words in passage and encodes the evidence relevant to the current passage word and its matching question information into the passage representation h_t^P:

$$h_t^P = \text{BiRNN}(h_{t-1}^P, [v_t^P, c_t]) \tag{7}$$

where $c_t = att(v^P, v_t^P)$ is an attention-pooling vector of the whole passage (v^P):

$$s_j^t = \text{v}^{\text{T}} \tanh(W_v^P v_j^P + W_v^{\tilde{P}} v_t^P)$$
$$a_i^t = \exp(s_i^t) / \Sigma_{j=1}^n \exp(s_j^t)$$
$$c_t = \Sigma_{i=1}^n a_i^t v_i^P \tag{8}$$

An additional gate as in gated attention-based recurrent networks is applied to $[v_t^P, c_t]$ to adaptively control the input of RNN.

4.1 Evidence Extraction: R-NET[5] and S-NET[8]...

- Output Layer:

h_j^P from eq 9 are v_j^P in S-NET from its previous layer as self-matching layer is excluded from S-NET

3.4 Output Layer

We follow Wang & Jiang (2016b) and use pointer networks (Vinyals et al., 2015) to predict the start and end position of the answer. In addition, we use an attention-pooling over the question representation to generate the initial hidden vector for the pointer network. Given the passage representation $\{h_t^P\}_{t=1}^n$, the attention mechanism is utilized as a pointer to select the start position (p^1) and end position (p^2) from the passage, which can be formulated as follows:

$$
\begin{aligned}
s_j^t &= \mathrm{v}^{\mathrm{T}} \tanh(W_h^P h_j^P + W_{\mathrm{h}}^a h_{t-1}^a) \\
a_i^t &= \exp(s_i^t) / \Sigma_{j=1}^n \exp(s_j^t) \\
p^t &= \mathrm{argmax}(a_1^t, \ldots, a_n^t)
\end{aligned} \tag{9}
$$

Here h_{t-1}^a represents the last hidden state of the answer recurrent network (pointer network). The input of the answer recurrent network is the attention-pooling vector based on current predicted probability a^t:

$$
\begin{aligned}
c_t &= \Sigma_{i=1}^n a_i^t h_i^P \\
h_t^a &= \mathrm{RNN}(h_{t-1}^a, c_t)
\end{aligned} \tag{10}
$$

When predicting the start position, h_{t-1}^a represents the initial hidden state of the answer recurrent network. We utilize the question vector r^Q as the initial state of the answer recurrent network. $r^Q = att(u^Q, V_r^Q)$ is an attention-pooling vector of the question based on the parameter V_r^Q:

$$
\begin{aligned}
s_j &= \mathrm{v}^{\mathrm{T}} \tanh(W_u^Q u_j^Q + W_{\mathrm{v}}^Q V_r^Q) \\
a_i &= \exp(s_i) / \Sigma_{j=1}^m \exp(s_j) \\
r^Q &= \Sigma_{i=1}^m a_i u_i^Q
\end{aligned} \tag{11}
$$

4.1 Evidence Extraction: R-NET[5] and S-NET[8]...

- Loss: The loss function is softmax cross-entropy and is defined as:

For this part, the objective function is to minimize the following cross entropy:

$$\mathcal{L}_{AP} = -\Sigma_{t=1}^{2}\Sigma_{i=1}^{N}[y_i^t \log a_i^t + (1 - y_i^t)\log(1 - a_i^t)] \quad (9)$$

where $y_i^t \in \{0, 1\}$ denotes a label. $y_i^t = 1$ means i is a correct position, otherwise $y_i^t = 0$.

4.2 Passage Ranking: S-NET[8]

3.2.2 Passage Ranking

In this part, we match the question and each passage from word level to passage level. Firstly, we use the question representation r^Q to attend words in each passage to obtain the passage representation r^P where $r^P = att(v^P, r^Q)$.

$$
\begin{aligned}
s_j &= \mathrm{v}^{\mathrm{T}} \tanh(W_v^P v_j^P + W_v^Q r^Q) \\
a_i &= \exp(s_i) / \Sigma_{j=1}^{n} \exp(s_j) \\
r^P &= \Sigma_{i=1}^{n} a_i v_i^P
\end{aligned} \tag{10}
$$

Next, the question representation r^Q and the passage representation r^P are combined to pass two fully connected layers for a matching score,

$$
g = v_g^{\mathrm{T}}(\tanh(W_g[r^Q, r^P])) \tag{11}
$$

For one question, each candidate passage P_i has a matching score g_i. We normalize their scores and optimize following objective function:

$$
\hat{g}_i = \exp(g_i) / \Sigma_{j=1}^{k} \exp(g_j)
$$

$$
\mathcal{L}_{PR} = -\sum_{i=1}^{k} [y_i \log \hat{g}_i + (1 - y_i) \log(1 - \hat{g}_i)] \tag{12}
$$

where k is the number of passages. $y_i \in \{0, 1\}$ denotes a label. $y_i = 1$ means P_i is the correct passage, otherwise $y_i = 0$.

4.2 Evidence Extraction: Joint Learning

3.2.3 Joint Learning

The evident extraction part is trained by minimizing joint objective functions:

$$\mathcal{L}_E = r\mathcal{L}_{AP} + (1 - r)\mathcal{L}_{PR} \tag{13}$$

where r is the hyper-parameter for weights of two loss functions.

4.3 Answer Synthesis: S-NET[8]:

3.3 Answer Synthesis

As shown in Figure 3, we use the sequence-to-sequence model to synthesize the answer with the extracted evidences as features. We first produce the representation h_t^P and h_t^Q of all words in the passage and question respectively. When producing the answer representation, we combine the basic word embedding e_t^p with additional features f_t^s and f_t^e to indicate the start and end positions of the evidence snippet respectively predicted by evidence extraction model. $f_t^s = 1$ and $f_t^e = 1$ mean the position t is the start and end of the evidence span, respectively.

$$h_t^P = \text{BiGRU}(h_{t-1}^P, [e_t^p, f_t^s, f_t^e])$$
$$h_t^Q = \text{BiGRU}(h_{t-1}^Q, e_t^Q) \quad (14)$$

4.3 Answer Synthesis: S-NET[8]...

On top of the encoder, we use GRU with attention as the decoder to produce the answer. At each decoding time step t, the GRU reads the previous word embedding w_{t-1} and previous context vector c_{t-1} as inputs to compute the new hidden state d_t. To initialize the GRU hidden state, we use a linear layer with the last backward encoder hidden state $\overleftarrow{h}_1^P$ and $\overleftarrow{h}_1^Q$ as input:

$$d_t = \text{GRU}(w_{t-1}, c_{t-1}, d_{t-1})$$
$$d_0 = \tanh(W_d[\overleftarrow{h}_1^P, \overleftarrow{h}_1^Q] + b) \quad (15)$$

where W_d is the weight matrix and b is the bias vector.

The context vector c_t for current time step t is computed through the concatenate attention mechanism (Luong et al., 2015), which matches the current decoder state d_t with each encoder hidden state h_t to get the weighted sum representation. Here h_i consists of the passage representation h_t^P and the question representation h_t^Q.

$$s_j^t = v_a^{\mathrm{T}} \tanh(W_a d_{t-1} + U_a h_j)$$
$$a_i^t = \exp(s_i^t) / \Sigma_{j=1}^n \exp(s_j^t)$$
$$c_t = \Sigma_{i=1}^n a_i^t h_i \quad (16)$$

4.3 Answer Synthesis: S-NET[8]...

We then combine the previous word embedding w_{t-1}, the current context vector c_t, and the decoder state d_t to construct the readout state r_t. The readout state is then passed through a maxout hidden layer (Goodfellow et al., 2013) to predict the next word with a softmax layer over the decoder vocabulary.

$$
\begin{aligned}
r_t &= W_r w_{t-1} + U_r c_t + V_r d_t \\
m_t &= [\max\{r_{t,2j-1}, r_{t,2j}\}]^{\mathrm{T}} \\
p(y_t|y_1, \ldots, y_{t-1}) &= \mathrm{softmax}(W_o m_t)
\end{aligned} \tag{17}
$$

where W_a, U_a, W_r, U_r, V_r and W_o are parameters to be learned. Readout state r_t is a $2d$-dimensional vector, and the maxout layer (Equation 17) picks the max value for every two numbers in r_t and produces a d-dimensional vector m_t.

Our goal is to maximize the output probability given the input sentence. Therefore, we optimize the negative log-likelihood loss function:

$$
\mathcal{L}_S = -\frac{1}{|\mathcal{D}|} \Sigma_{(X,Y)\in\mathcal{D}} \log p(Y|X) \tag{18}
$$

where $\mathcal{D}$ is the set of data. X represents the question and passage including evidence snippets, and Y represents the answer.

5. Experiment: R-NET[5]

Framework: Python 3, TensorFlow 1.4.0
Loss: Softmax cross entropy
Optimizer: Adadelta with initial learning rate 0.5
Dropout: 0.3
Hidden Dimension: 75
Training time (on 8GB GTX 1080 Ti GPU) before it converges:
    7 hours (SQuAD, 60k iterations, approx 50 epochs)
    21 hours (MS-MARCO, 80k iterations, approx 18 epochs)
R-NET on SQuAD implementation source[7]: https://github.com/HKUST-KnowComp/R-Net/

5. Experiment: S-NET[8]

Framework: Python 3, TensorFlow 1.4.0
Dataset: MS-MARCO
Loss: Softmax cross entropy
Optimizer: Adadelta with initial learning rate: 1.0
Dropout: 0.1
Hidden Dimension: 150
Training time (on 8GB GTX 1080 Ti GPU) before it converges:

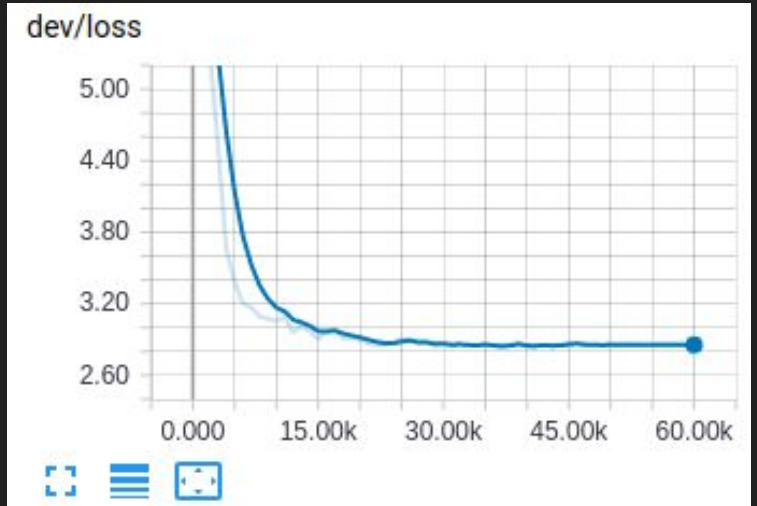
- 10+ days (Evidence Snippet Prediction, 49k iterations, 16 batch size)
- 4+ days (ESP with Passage Ranking, 18k iterations, 8 batch size)
- 5 hours (Answer Synthesis, 12k iterations, 64 batch size)

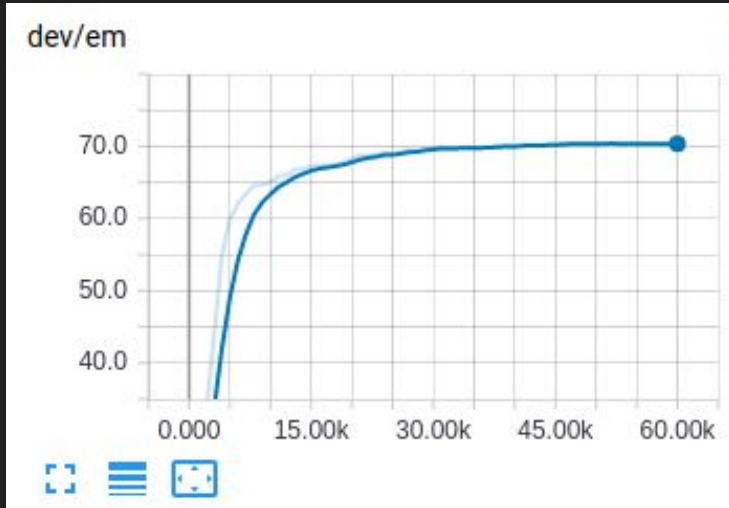
S-NET on MS-MARCO implementation source[7,9]:
https://github.com/HKUST-KnowComp/R-Net/, https://github.com/tensorflow/nmt/tree/master/nmt

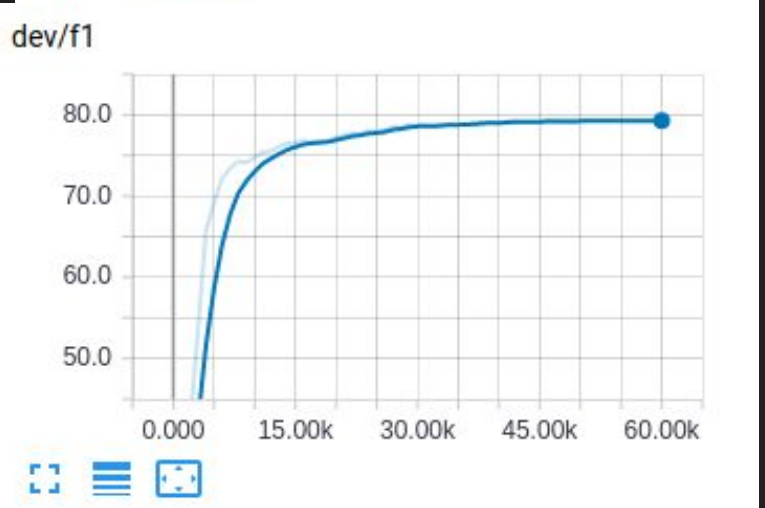
6.1.1. Results (R-NET: SQuAD)

Batch size: 64

Dev Loss, EM (Exact Matching) and F1 score:



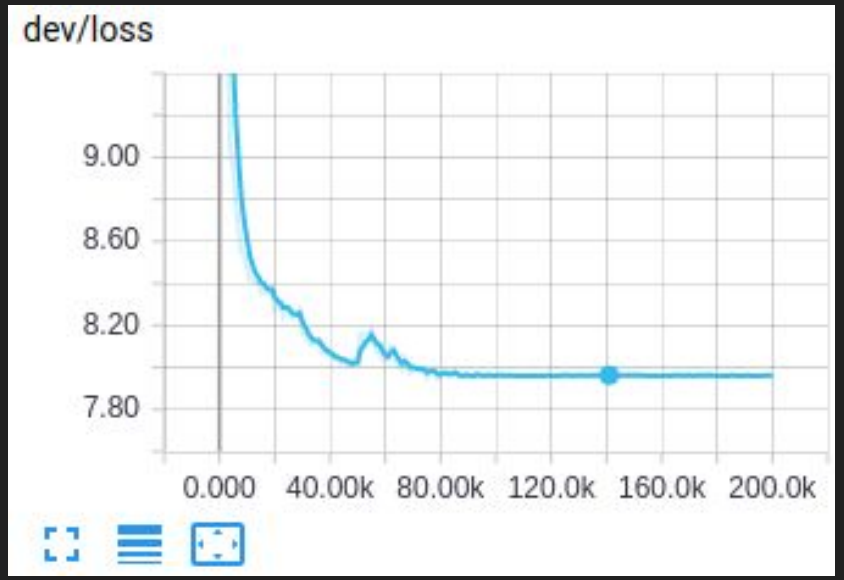


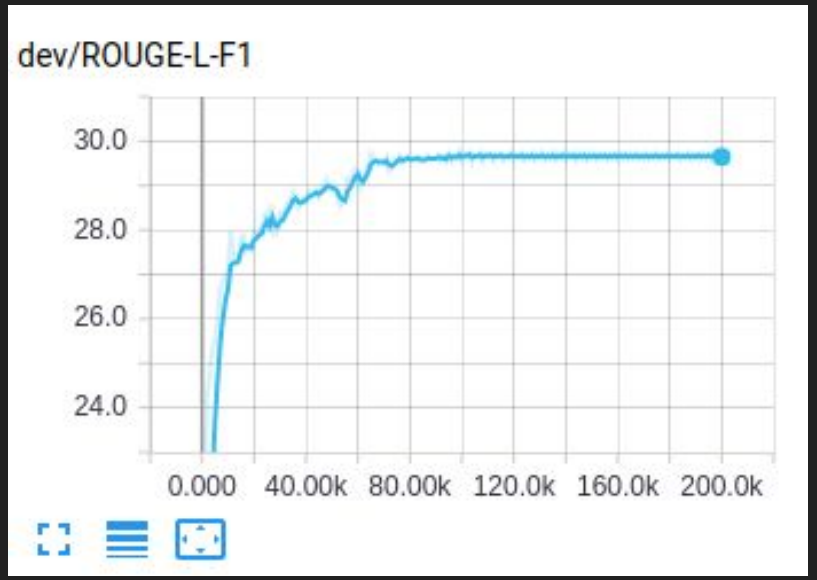


6.1.2. Results (R-NET: MS-MARCO)

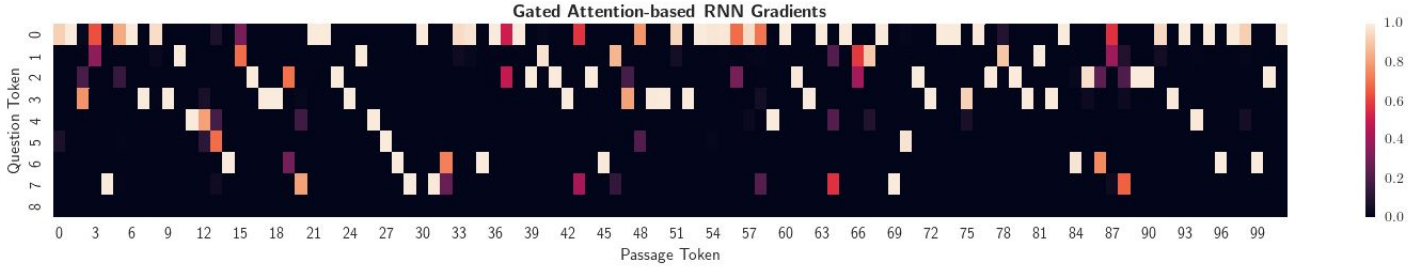
Batch size: 16

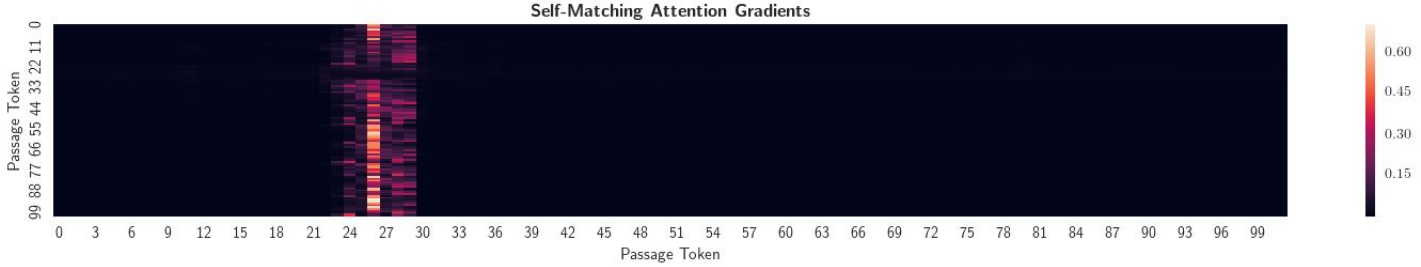
Dev Loss, and ROUGE-L-F1 score: (Paper: 42.9)

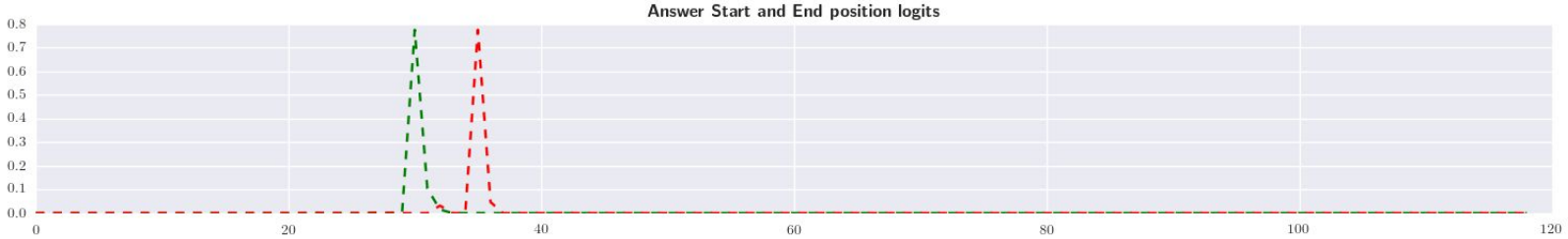




6.1. Results: Attention gradients and start/end position heatmap



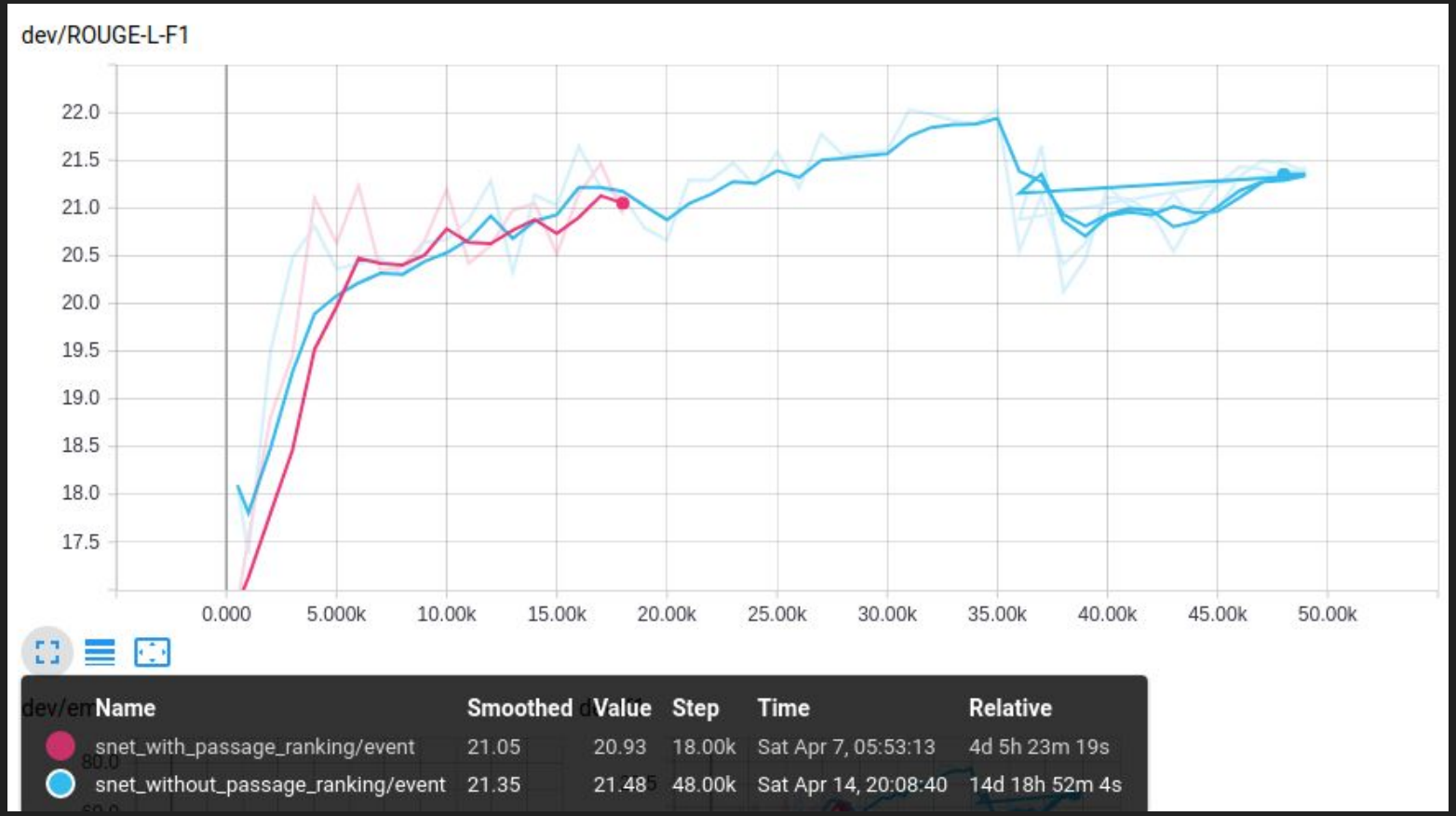




Question: ['How', 'is', 'unregistered', 'property', 'held', 'in', 'informal', 'form', '?']

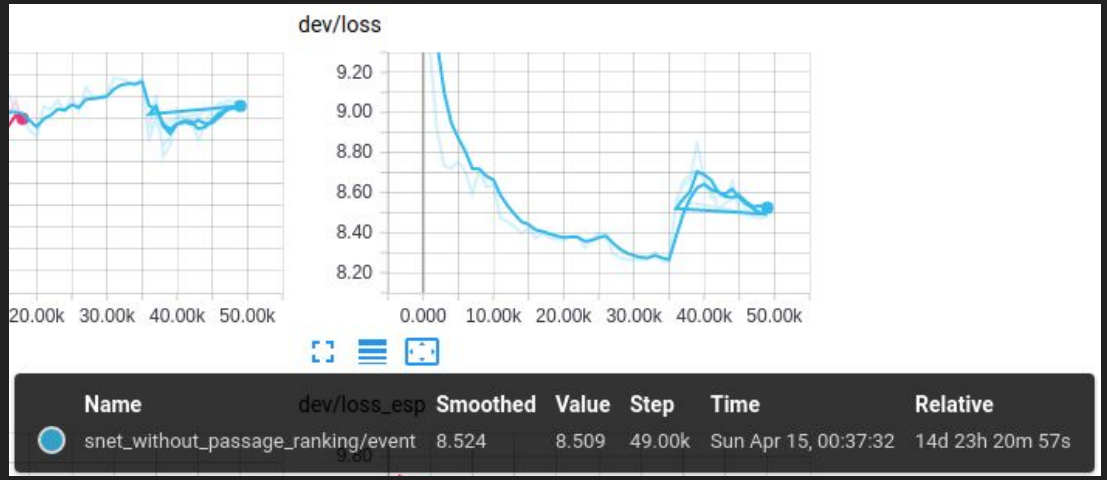
22 : Much
23 : unregistered
24 : property
25 : is
26 : held
27 : in
28 : informal
29 : form
30 : through
31 : various
32 : associations
33 : and
34 : other
35 : arrangements
36 : .
37 :

6.2.1 Results: S-NET (with and w/o Passage Ranking)

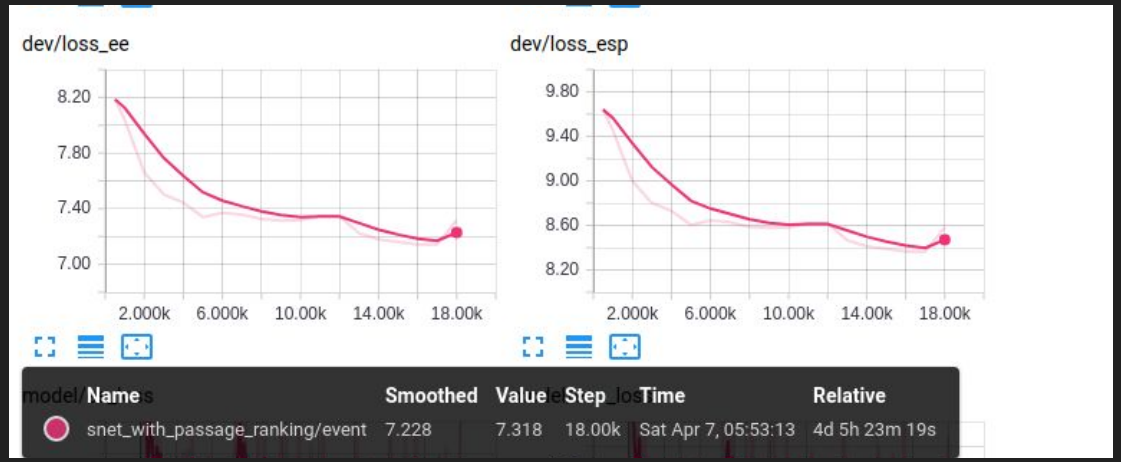


6.2.1 Results: S-NET loss (with and w/o Passage Ranking)

S-NET Loss: Without passage ranking:



S-NET Loss: With Passage Ranking



6.2.2 Results: S-NET (Answer Synthesis)

- Results using features and training data directly from the dataset:
  63.0 ROUGE-L score
- Results using features and training data from the Evidence Extraction outputs:
  15.3 ROUGE-L score (Paper score: 44)

Reasons: Poor outputs from Evidence Extraction which is again due to premature convergence of loss during training.

Method	ROUGE-L paper / ours
RNET (Single Para)	N.A/51.46
RNET (Multi Para)	42.89/27.93
S-NET (w/o Passage Ranking)	39.62/20.30
S-NET (Extraction)	42.23/19.87
S-NET	47.76/15.3

TABLE 1: Our performance on the MS-MARCO dev dataset

References

[1] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, Percy Liang. SQuAD: 100,000+ Questions for Machine Comprehension of Text. arXiv:1606.05250
[2] Tri Nguyen, Mir Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, Li Deng. MS MARCO: A Human Generated MAchine Reading COmprehension Dataset. arXiv:1611.09268
[3] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. GloVe: Global Vectors for Word Representation.
[4] Chin-Yew Lin. ROUGE: A Package for Automatic Evaluation of Summaries. http://www.aclweb.org/anthology/W04-1013
[5] Natural Language Computing Group, Microsoft Research Asia. R-NET: MACHINE READING COMPREHENSION SELF -MATCHING NETWORKS
[6] Tim Rocktäschel, Edward Grefenstette, Karl Moritz Hermann, Tomáš Kočiský, Phil Blunsom. Reasoning about Entailment with Neural Attention. arXiv:1509.06664
[7] R-NET on SQuAD: https://github.com/HKUST-KnowComp/R-Net/
[8] Chuanqi Tan, Furu Wei, Nan Yang, Bowen Du, Weifeng Lv, Ming Zhou. S-NET: From Answer Extraction to Answer Generation for Machine Reading Comprehension. arXiv:1706.04815
[9] Neural Machine Translation: https://github.com/tensorflow/nmt/tree/master/nmt

THANK YOU