Alpha and Omega? Influence of Answer-Pointer Frameworks in Question Answering Models

CS 224N, Winter 2022

Eden Grown-Haeberli and Nicholas Paul-Brazeau Sanchez

Problem

Question Answering: In reading comprehension, a paragraph and a question about the paragraph is provided to a model as input and an answer to the question is provided as output.

Question answering is important for three reasons:

- Almost all NLP tasks can be posed as question answering problems
- Question answering is the first step to reading comprehension
- NLP tasks and therefore QA tasks form the basis of most human computer interactions

Background

SQUaD Dataset:

- Paragraphs from Wikipedia and 150k questions
- Questions and answers crowdsourced using Turk
- Half of the questions cannot be answered using paragraph

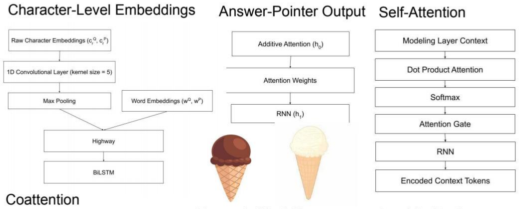
Often, successful SQUaD models condition the end token on the start token. Answer-Pointer generates probability distribution based on attention across context tokens. Answer-Pointer derivatives include:

- R-Net: Machine Reading Comprehension with Self-Matching Networks
- Dynamic Coattention Networks for Question Answering

Which models should apply an Answer-Pointer layer in order to improve performance? Answer-Pointer output might be too deep or sophisticated for the size of the dataset.

Goal: Save time for future modelers and have a benchmark for which architectures benefit from Answer-Pointer layers by examining the effect of an Answer-Pointer output on the performance of the model.

Methods



Character-Level Embeddings

Raw Character Embeddings (c_r^Q, c_r^P)

1D Convolutional Layer (kernel size = 5)

Max Pooling

Word Embeddings (w^Q, w^P)

Highway

BiLSTM

Answer-Pointer Output

Additive Attention (h_0)

Attention Weights

RNN (h_1)

Self-Attention

Modeling Layer Context

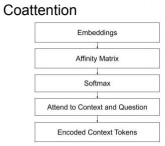
Dot Product Attention

Softmax

Attention Gate

RNN

Encoded Context Tokens



Coattention

Embeddings

Affinity Matrix

Softmax

Attend to Context and Question

Encoded Context Tokens

Figure 1. Model improvements added to the baseline BiDAF model. Improvements were mixed and matched according to the ablation testing schedule described in methods. Answer-pointer output layer was substituted in for BiDAF output layer and the models were compared.

Ablation testing to compare models:

- A: Baseline BiDAF
- B: Char embeddings + BiDAF
- C: Char embeddings + self-attention + BiDAF
- D: Char embeddings + coattention + BiDAF
- E: Self-attention + BiDAF

A’: Baseline with Answer-Pointer output (AP)

B’: Character embeddings + AP

C’: Char embeddings + self-attention + AP

D’: Char embeddings + coattention + AP

Results

Model	EM	F1
Model A	57.13	60.67
Model A’	58.34	61.82
ΔA	**+1.21**	**+1.15**
Model B	61.44	64.72
Model B’	61.07	64.31
ΔB	**-0.34**	**-0.41**
Model C	59.77	63.59
Model C’	62.06	65.33
ΔC	**+2.29**	**+1.74**
Model D	59.94	63.4
Model D’	59.23	63.06
ΔD	**-0.71**	**-.34**

Model	EM	F1
Model A’	58.34	60.67
Model A”	55.82	59.07
Δ	**-2.52**	**-1.67**
Model C’	62.06	65.33
Model C”	61.62	64.84
Δ	**-0.44**	**-0.49**

Model	EM	F1
Model A	57.13	60.67
Model E	59.18	62.24
Δ	**+2.05**	**+1.57**

Test Leaderboard Results:

EM: 61.927
F1: 65.560

Figure 2. Results for models A, A’, A’’, B, B’, C, C’, C’’, D, D’, E

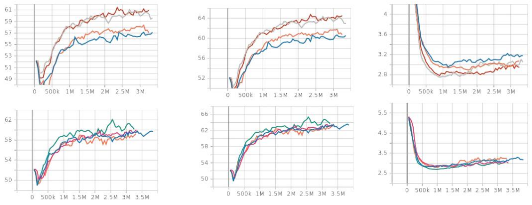


Figure 3. From left to right: EM, F1, and NLL; Top: A: Blue, A': Orange, B: Red, B': Grey; Bottom: C: Blue, C': Green, D: Pink, D': Orange

B Models:

- Answer-pointer layer detracts from B model
- Expected improvement to be orthogonal to the morphology level benefit provided by character embeddings

C Models:

- Answer-pointer layer and self-attention both rely on self-attention over context tokens and answer-pointer layer improves C model

D Models:

- D’ able to extract correct interpretation in some cases despite having worse overall performance (see Fig 4. below)

- **Question:** What equals the spring reaction force on an object suspended on a spring reaction scale?
- **Answer:** Gravity
- **C’ Prediction:** equals the object’s weight
- **D’ Prediction:** gravity acting

Fig 4. Example of QA pair from C’ and D’

Analysis

Qualitative Observations about Answer-Pointer Models vs BiDAF Output Models

- Better at encapsulating articles like “the” or “and”
- Selects more verbose answers
- More biased against giving N/A as an answer
- Sensitive to irregular punctuation schemas
  - **Question:** What type of group is The Islamic State?
  - **Context:** "The islamic State"...is a Wahhabi/Salafi jihadist extremist militant group...
  - **Answer:** Wahhabi/Salafi jihadist extremist militant
  - **C Prediction:** extremist militant
  - **C’ Prediction:** Wahhabi/Salafi jihadist extremist militant

Fig 5. Example of QA pair from C/C’.

Answer-Pointer Specific Qualitative Explanations

- Attention based categorical distributions over start and end tokens with argmax of these as start and end tokens output
- Very concentrated attention at the start token suggests that the passage has an answer
- Uses attention knowledge to predict less likely tokens from the context causing end token attention distributions to become more concentrated
- OOV token has less attention weight by comparison in the end token layer making model less likely to predict N/A
- Understands text well when the question text is directly in the context paragraph or the wording is close to the input question
- Does not understand well when question text incorporates a word or abbreviation that the model has not seen

Conclusions

Model Observations

- Chocolate answer pointer provided inconsistent benefit to models
- Conject that B, C, and D already capture information that answer pointer could provide, leading to answer pointer effects being dampened in B’, C’, D’

Nest steps

- Propose aggressively regularizing answer pointer RNN (perhaps applying dropout on start token inputs)
- Explore whether C underperforms C’ with different modeling seeds

References

N. L. C. Group, “R-net: Machine reading comprehension with self-matching networks,” May 2017. [Online]. Available: https://www.microsoft.com/en-us/research/publication/mcr/

M.J. Seo, A. Kembhavi, A. Farhadi, and H. Hajishirzi,“Bidirectional attention flow for machine comprehension,” CoRR, vol. abs/1611.01603, 2016. [Online]. Available: http://arxiv.org/abs/1611.01603

S. Wang and J. Jiang, “Machine comprehension using match-lstm and answer pointer,” CoRR, vol. abs/1608.07905, 2016. [Online]. Available: http://arxiv.org/abs/1608.07905

C. Xiong, V. Zhong, and R. Socher, “Dynamic coattention networks for question answering,” 2018.

P. Rajpurkar, J. Zhang, K. Lopyrev, and P. Liang, “Squad: 100,000+ questions for machine comprehension of text,” CoRR, vol. abs/1606.05250, 2016. [Online]. Available: http://arxiv.org/abs/1606.05250

O. Vinyals, M. Fortunato, and N. Jaitly, “Pointer networks,” in NIPS, 2015, pp. 2692–2700. [Online]. Available: https://arxiv.org/pdf/1506.03134.pdf

C. staff. Cs 224n default final project: Building a qa system (iid squad track). [Online]. Available: http://web.stanford.edu/class/cs224n/project/default-final-project-handout-squad-track.pdf

A. Petrushko. R-net-pytorch/outputlayer.py at a6ed4a02b0cf68bade9e9a43a93ec290a3b6fabdtailerr/r-net-pytorch. [Online]. Available: https://github.com/tailerr/R-NET-pytorch/blob/a6ed4a02b0cf68bade9e9a43a93ec290a3b6fabd/source/nn/modules/output_layer.py

Slashcom, “R/pytorch-pytorch multi-head attention module.” [Online]. Available: https://reddit.com/r/pytorch/comments/c2u6g5/pytorch_multihead_attention_module