Token-level Dynamic Self-Attention Network for Multi-Passage Reading Comprehension

Yimeng Zhuang, Huadong Wang
Samsung Research China - Beijing (SRC-B)
{ym.zhuang, huadong.wang}@samsung.com

Abstract

Multi-passage reading comprehension requires the ability to combine cross-passage information and reason over multiple passages to infer the answer. In this paper, we introduce the Dynamic Self-attention Network (DynSAN) for multi-passage reading comprehension task, which processes cross-passage information at token-level and meanwhile avoids substantial computational costs. The core module of the dynamic self-attention is a proposed gated token selection mechanism, which dynamically selects important tokens from a sequence. These chosen tokens will attend to each other via a self-attention mechanism to model long-range dependencies. Besides, convolutional layers are combined with the dynamic self-attention to enhance the model's capacity of extracting local semantic. The experimental results show that the proposed DynSAN achieves new state-of-the-art performance on the SearchQA, Quasar-T and WikiHop datasets. Further ablation study also validates the effectiveness of our model components.

词级别的动态
self-attention,
避免大量的计算损失

评估

1 Introduction

As a critical approach for evaluating the ability of an intelligent agent to understand natural language, reading comprehension (RC) is a challenging research direction, attracting many researchers' interest. In real application scenarios, such as web search, the passages may be multiple and extended, and may be comprised of relevant and irrelevant contents. It involves the problem of multi-passage reading comprehension.

In multi-passage setting, cross-passage information interaction is vital for modeling long-range dependencies, co-references between entities in different passages (Dhingra et al., 2018), cross-passage answer verification (Wang et al., 2018b), and multihop reasoning (Welbl et al., 2018), etc.

Great efforts have been made to develop models for multi-passage task, such as Wang et al. (2018b); Zhong et al. (2019); Dehghani et al. (2019a); Dhingra et al. (2018); De Cao et al. (2019); Song et al. (2018). The common practice of these approaches is that all the embeddings in a passage or a span are integrated into a single vector and the cross-passage information interactions are based on these coarse-grain semantic representations. However, it may cause potential issues. As is pointed out in Bahdanau et al. (2015); Cho et al. (2014), compressing all the necessary information into a single vector may lead to "sacrifice" some critical information due to the allocated capacity to remember other information. This problem is prevalent in Neural Machine Translation (NMT), the recent models, such as the Transformer (Vaswani et al., 2017), workaround this issue by decoding on token-level context encodings of the source text. As such, we hypothesize that fine-grain representations may keep precise semantic information, and may be beneficial to cross-passage information interactions in RC tasks. In this paper, we focus on an architecture which deals with the cross-passage information at token-level.

The proposed architecture is a variant of the Self-attention Network (SAN) (Vaswani et al., 2017; Shen et al., 2018a). Our model employs a self-attention mechanism to combine token-level supportive information from all passages in a multi-step process. Directly applying self-attention over all tokens is computationally expensive. Instead, in each step, the most important tokens are dynamically selected from all passages, and information interaction only happens over these chosen tokens via the self-attention mechanism. The motivation behind it is an observation that the information used to answer the question is usually concentrated on a few words.

Our experiments verify this observation to a certain extent. We expect that our model can automatically find out these important tokens. Thus we propose a gated token selection mechanism and equip it with the self-attention module.

We intend the model to achieve a balance in speed, memory, and accuracy. While the self-attention mechanism is widely used in end-to-end models to capture long-range dependency, it is intrinsically inefficient in memory usage. Shen et al. (2018b) elaborates the memory issue. The memory required to store the attention matrix grows quadratically with the sequence length. Considering real scenarios, such as web search, in which the retrieval system returns hundreds of articles, and each contains hundreds or thousands of words, thus applying self-attention on all tokens in the supporting passages is computationally expensive. Compared to recurrent neural networks, such as LSTM (Hochreiter and Schmidhuber, 1997), SAN is highly parallelizable and usually faster on long sequence (Vaswani et al., 2017). The proposed method accomplishes necessary cross-passage information interaction with a time/memory complexity linear in the length of the sequence and do not add much extra calculation burden.

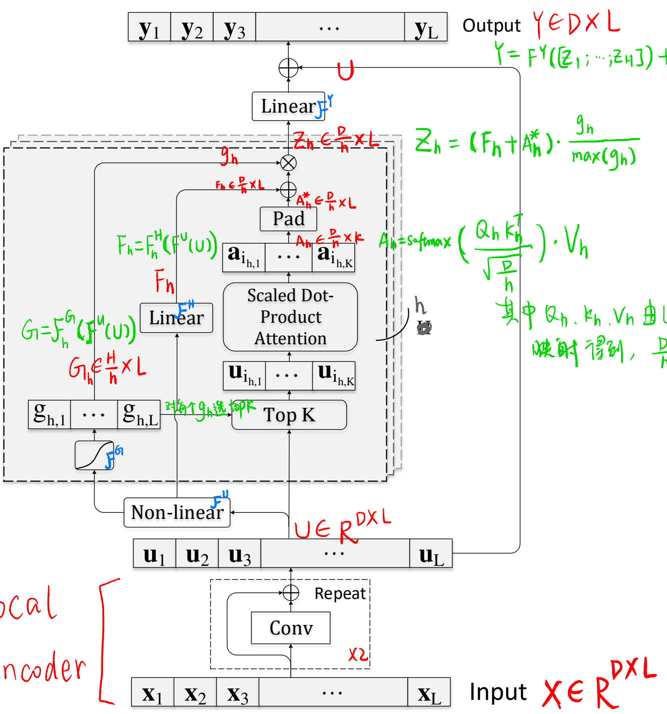
Our contributions in this work are as follows: (1) We propose Dynamic Self-attention (DynSA) for information interaction in a long sequence. (2) Token-level cross-passage information interaction is implemented through the application of the proposed DynSA at relatively less computational costs. (3) Our Dynamic Self-attention Network (DynSAN) achieves new state-of-the-art performance compared with previously published results on SearchQA, Quasar-T and WikiHop benchmarks.

2 Dynamic Self-attention Block

This section introduces the Dynamic Self-Attention Block (DynSA Block), which is central to the proposed architecture. The overall architecture is depicted in Figure 1.

核心思想：
gated token selection
评估每个词的重要性，选出topk重要的词.
self-attention机制仅作用在选出的topk词上.

The core idea of this module is a gated token selection mechanism and a self-attention. We expect that a gate can acquire the estimation of each token's importance in an input sequence, and use this estimated importance to extract the most important K tokens. Then we run a self-attention, instead of computing the full self-attention matrix over all the tokens, only the chosen K tokens are



$\mathbf{y}_1$ $\mathbf{y}_2$ $\mathbf{y}_3$ ⋯ $\mathbf{y}_L$
Output $Y \in D \times L$
$Y = F^Y([Z_1; \cdots; Z_H]) + U$
U
Linear F^Y
$Z_h \in \frac{D}{h} \times L$
$Z_h = (F_h + A_h^*) \cdot \frac{g_h}{\max(g_h)}$
g_h
$F_h \in \frac{D}{h} \times L$
$A_h^* \in \frac{D}{h} \times L$
Pad
$A_h \in \frac{D}{h} \times K$
$A_h = \mathrm{softmax}\left(\frac{Q_h K_h^T}{\sqrt{\frac{D}{h}}}\right) \cdot V_h$
$F_h = F_h^H(F^U(U))$
$\mathbf{a}_{i_{h,1}}$ ⋯ $\mathbf{a}_{i_{h,K}}$
F_h
Linear F^H
Scaled Dot-Product Attention
h
其中 Q_h, K_h, V_h 由 U_h 线性映射得到，$\frac{D}{h} \times L$.
$G = F_h^G(F^U(U))$
$G_h \in \frac{H}{h} \times L$
$\mathbf{u}_{i_{h,1}}$ ⋯ $\mathbf{u}_{i_{h,K}}$
$\mathbf{g}_{h,1}$ ⋯ $\mathbf{g}_{h,L}$
对每个 g_h 选topK
Top K
F^G
Non-linear F^U
$U \in R^{D \times L}$
$\mathbf{u}_1$ $\mathbf{u}_2$ $\mathbf{u}_3$ ⋯ $\mathbf{u}_L$
Local Encoder
Repeat
Conv
x2
$\mathbf{x}_1$ $\mathbf{x}_2$ $\mathbf{x}_3$ ⋯ $\mathbf{x}_L$
Input $X \in R^{D \times L}$

Figure 1: Architecture of the Dynamic Self-Attention Block.

taken into account. This module results in lower memory consumption and makes the self-attention focus on the active part of a long input sequence. The above idea is implemented through stacking two structures: a local encoder and a dynamic self-attention module. 模块组成

2.1 Local Encoder

In the architecture, a local encoder is used to encode local information, such as short-range context, which is useful for disambiguation. The reasons for the local encoder are that (1) only computing self-attention over a few tokens among a long sequence may lead the self-attention to lose the capability of modeling short-range context for every position in the sequence, and (2) after a position receives the attended information from long-range positions, the local encoder is needed to spread this information to its neighboring positions, and (3) previous works have proven that combining a local encoder with self-attention is beneficial in some tasks (Yu et al., 2018).

A natural candidate for the local encoder is local convolution, which is widely used as local feature extractors. Besides, restricted self-attention (Vaswani et al., 2017) is also a choice. In this work, we adopt 1D convolution as the local encoder. Specifically, let $\boldsymbol{X} \in \mathbb{R}^{D \times L}$ be the in-

depth-wise separable conv.
设卷积核宽度为k，则核的大小为 $1 \times k$，而不是 $D \times k$.

put matrix of an L-token sequence, and each token embedding is D-dimensional. The output of a convolutional layer is calculated with a residual connection: $Conv(LN(\boldsymbol{X})) + \boldsymbol{X}$, where LN is the layer normalization (Ba et al., 2016), $Conv$ denotes a convolutional layer. For less computational costs, we adopt depth-wise separable convolutions (Chollet, 2017) throughout this paper. The local encoder consists of a stack of 2 convolutional layers.

2.2 Dynamic Self-attention

Since our self-attention is performed over a set of tokens which are determined dynamically, we call it Dynamic Self-Attention (DynSA). The DynSA is based on the hypothesis that the number of important tokens is much less than the sequence length in a long sequence. Here, to say a token is important means that the token contains the necessary information to enable the model to predict the answer, or the token is non-negligible for modeling long-range semantic dependency. DynSA intends to find out the most important tokens by a token selection mechanism and then performs a self-attention only over these chosen tokens.

In DynSA, we use a gate to control how much of the output, which includes non-linear transformations and attended vectors, to pass this layer. A large gate activation value implies that the corresponding output is important in this layer. Thus, we use the gate activation as the basis of token selection. Given the output of the local encoder $\boldsymbol{U} \in \mathbb{R}^{D\times L}$, the gate activation is computed via:

仿射层+sigmoid

$$\boldsymbol{G} = \underline{\mathcal{F}^G}(\mathcal{F}^U(\boldsymbol{U})) \in R^{H\times L} \quad (1)$$

(代表有H个head)

链接+激活函数

where $\mathcal{F}^U$ denotes a non-linear fully connected layer, $\mathcal{F}^G$ denotes an affine transformation with $sigmoid$ activation function. In our work, we allow to use multi-head attention (Vaswani et al., 2017). Equation 1 outputs $\boldsymbol{G} \in \mathbb{R}^{H\times L}$, which contains H heads. And we use $\boldsymbol{g}_h \in \mathbb{R}^L$ (the h-th row in $\boldsymbol{G}$) to represent the gate output of the h-th head. The element $g_{h,i}$ in $\boldsymbol{g}_h$ is the gate activation corresponding to the token at the i-th position.

每个head选topK词.

Then, in each head we select the top K tokens according to their corresponding gate activations in $\boldsymbol{g}_h$, in which K is a hyper-parameter. In case of the actual sequence length being less than K, we select all the tokens. We get the chosen tokens' embeddings $\boldsymbol{U}_h = [\boldsymbol{u}_{i_{h,1}}, \cdots, \boldsymbol{u}_{i_{h,j}}, \cdots, \boldsymbol{u}_{i_{h,K}}] \in \mathbb{R}^{D\times K}$, where

第h个head选出的词.

$i_{h,j} \in \{1, 2, \cdots, L\}$ is the position index of the chosen token in the input sequence. We consider this as a gated token selection mechanism.

Scaled dot-product attention is adopted over the chosen tokens:

对Uh进行线性映射.

对选出的词进行self-attention

$$\boldsymbol{A}_h = \text{softmax}(\frac{\overbrace{\boldsymbol{Q}_h\boldsymbol{K}_h^T}^{\in \frac{D}{h}\times K}}{\sqrt{D/H}}) \cdot \boldsymbol{V}_h \quad (2)$$

where $\boldsymbol{Q}_h \in \mathbb{R}^{\frac{D}{H}\times K}$, $\boldsymbol{K}_h \in \mathbb{R}^{\frac{D}{H}\times K}$, and $\boldsymbol{V}_h \in \mathbb{R}^{\frac{D}{H}\times K}$ are query, key, and value respectively, they are linear projections of the input $\boldsymbol{U}_h$. $\boldsymbol{A}_h \in \mathbb{R}^{\frac{D}{H}\times K}$ is the attended output matrix of the h-th head.

Next, we pad those unchosen positions with zero embeddings to complete the sequence length. Having $\boldsymbol{A}_h^* = \text{Pad}(\boldsymbol{A}_h) \in \mathbb{R}^{\frac{D}{H}\times L}$. The output of the h-th head $\boldsymbol{Z}_h \in \mathbb{R}^{\frac{D}{H}\times L}$ is calculated as follows,

$$\boldsymbol{Z}_h = (\overset{\frac{D}{h}\times L}{\boldsymbol{F}_h} + \boldsymbol{A}_h^*) \cdot \frac{\overset{R^L}{\boldsymbol{g}_h}}{\max(\boldsymbol{g}_h)} \in \frac{D}{h}\times L \quad (3)$$

$\boldsymbol{g}_h$中最大的元素

$$\boldsymbol{F}_h = \underset{仿射层}{\mathcal{F}_h^H}(\mathcal{F}^U(\boldsymbol{U})) \in \frac{D}{h}\times L \quad (4)$$

where $\mathcal{F}_h^H$ is an affine layer. Equation 4 produces a non-linear transformation $\boldsymbol{F}_h \in \mathbb{R}^{\frac{D}{H}\times L}$ of the input embeddings. Since zero embeddings are padded at unchosen positions, by adding $\boldsymbol{F}_h$ gradient vanishing can be avoided when updating the parameters of the gate in training phase. In Equation 3, the maximum operation aims to select the maximum element in vector $\boldsymbol{g}_h$, and the division operation normalizes these elements so that the maximum activation is always one.

Finally, the output $\boldsymbol{Y} \in \mathbb{R}^{D\times L}$ of a DynSA block is the fusion of all heads.

$$\boldsymbol{Y} = \underset{线性映射}{\mathcal{F}^Y}(\underset{D\times L}{[\boldsymbol{Z}_1; \cdots; \boldsymbol{Z}_H]}) + \underset{D\times L}{\underline{\boldsymbol{U}}} \in D\times L \quad (5)$$

in which, $\mathcal{F}^Y$ denotes a linear projection, $[\cdot;\cdot]$ is the concatenation of the outputs of all heads.

Optionally, we suggest adding a regularization on the gate activation to make it more sparse, so that those unimportant tokens' activation values are almost zero and let the model generate more discriminative gate activation. Experiments show that the regularization can produce small gains in performance. Specifically, we jointly optimize the following regularization term when training the model.

$$\mathcal{L}^* = \beta \cdot ||\boldsymbol{G}||_1 \quad (6)$$

在训练中加上对G的正则化项.

where $\boldsymbol{G}$ represents the gate activation, $||\cdot||_1$ denotes 1-norm. β is a small hyper-parameter, which is set to 10^{-5} in our experiments.

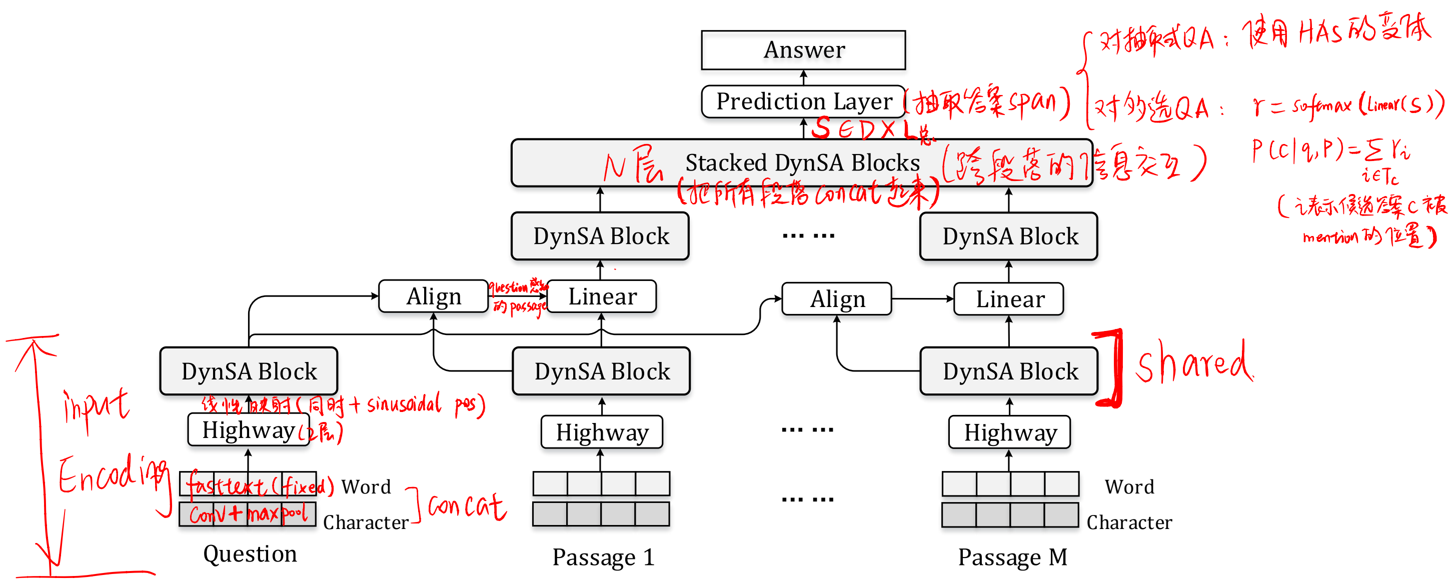


Figure 2: Architecture of Dynamic Self-Attention Network (DynSAN) for multi-passage reading comprehension.

3 Token-level Dynamic Self-attention Network

This section introduces the application of our proposed Dynamic Self-attention Network (DynSAN) on the multi-passage RC task. Given a question and M passages, it requires the model to predict a span from the passages to answer the question. Figure 2 illustrates the architecture of DynSAN.

3.1 Input Encoding

At the bottom of DynSAN, the input texts are first converted into distributional representations. We use the concatenation of word embeddings and character encodings for every single token. For word embedding, we adopt the pre-trained 300-dimensional fasttext Mikolov et al. (2018) word embeddings and fix them during training. Character encodings are obtained by performing convolution and max-pooling on 15-dimensional randomly initialized character embeddings following (Kim, 2014). Character embeddings are trainable while word embeddings are fixed in the training phase. On top of the embeddings, we adopt a 2-layer highway network (Srivastava et al., 2015) for deep transformation. The output of the highway network is immediately mapped to D dimensions through a linear projection, and we add sinusoidal positional embeddings (Vaswani et al., 2017) to the vectors for each token to expose position information to the model. Then, the vectors are fed into a layer of DynSA blocks. These DynSA Blocks are in charge of independently encoding context information inside the question and every passage, in which the parameters of DynSA blocks are shared in the layer. We use DynSA rather than the full multi-head self-attention to avoid massive memory consumption caused by exceptionally long passages.

3.2 Alignment

Alignment is a common and necessary step to generate question-aware context vectors for each passage, here, we adopt the strategy used in Yu et al. (2018), in which it includes a trilinear co-attention (Weissenborn et al., 2017) and a heuristic combination with query-to-context (Seo et al., 2017). ?? ??
Due to the limited space, we encourage reading the references for detailed descriptions and omit the repeated introduction. Then, the question-aware context vectors are projected into the standard dimension D through a linear layer and are encoded by a layer of DynSA blocks again to build semantic representations inside each passage further.

3.3 Cross-Passage Attention

Thus far, each passage aligns with the question independently, and DynSA blocks generate contextual embeddings inside each passage independently, so there is no interaction between passages. For multi-passage reading comprehension, cross-passage information interaction is beneficial to solve the problems, such as multihop reasoning, and multi-passage verification. Previous works either omit the cross-passage interaction (Clark and Gardner, 2018) or implement it at a relatively coarse granularity (Dehghani et al., 2019a). For example, in Dehghani et al. (2019a), each passage is encoded into a singular vector and self-attention is performed over these passage vectors. Instead of passage-level or block-level interaction (Shen et al., 2018b), in this work, we focus on modeling cross-passage long-range dependencies at token-level through a cross-passage attention layer. We expect that fine-grain self-attention may keep precise semantic information.

This layer consists of N stacked DynSA blocks. Specifically, as is shown in Figure 2, we concatenate the vector sequences of all passages end to end, and then stack N layers of DynSA blocks on top of this long vector sequence. If these passages are given in order, for instance, the passages have been ranked by a search engine, we add a rank embedding to each passage before the concatenation. The rank embeddings are randomly initialized, and the i-th rank embedding is added to every token vector in the i-th ranked passage.

3.4 Prediction Layer

The prediction layer is used to extract the answer span based on the output of previous layers. Depend on the type of tasks, different architectures are chosen. In this work, we investigate extractive QA and multiple choice QA.

3.4.1 Extractive QA

Extractive QA is challenging since we have to extract the answer span from the passages without any given candidate answer. In this paper, we adopt the Hierarchical Answer Spans (HAS) model (Pang et al., 2019) to solve this problem. Details are included in Pang et al. (2019), and we do not repeat it here due to limited space. In our implementation, the differences to Pang et al. (2019) are that the start/end probability distribution is calculated over all tokens as in Equation 7, RNN is replaced with DynSA block, and the paragraph quality estimator mentioned in Pang et al. (2019) is not used.

3.4.2 Multiple Choice QA

In this type of task, a list of candidate answers is provided. Here, we assume $\boldsymbol{S} \in \mathbb{R}^{D \times L}$ as the output of the cross-passage attention layer, L represents the total length of the M passages, q denotes the question, and $P = \{p_1, \cdots, p_M\}$ denotes the set of passages. We first convert the token vectors into a probability distribution $\boldsymbol{r} \in \mathbb{R}^L$ over all tokens,

$$\boldsymbol{r} = \mathrm{softmax}(\boldsymbol{\mathcal{F}}^S(\boldsymbol{S})) \tag{7}$$

where $\boldsymbol{\mathcal{F}}^S$ is a linear projection.

The probability of choosing a candidate c as the answer is computed via:

$$P(c|q, P) = \sum_{i \in \mathcal{T}_c} r_i \tag{8}$$

where $\mathcal{T}_c$ is a set of positions where the candidate c's mentions appear. During training, we optimize the log-likelihood of choosing the correct answer's probability.

4 Experiments

4.1 Datasets

We conduct experiments to study the performance of the proposed approach on three publicly available multi-passage RC datasets.

SearchQA (Dunn et al., 2017) is an open domain QA dataset including about 140k questions crawled from J! Archive, and about 50 web page snippets, which are retrieved from the Google search engine, as the supporting passages for each question. The authors of SearchQA have provided a processed version of this dataset, in which all words are lower-cased, and tokenization has been completed. Our experiments are based on this processed version.

Quasar-T (Dhingra et al., 2017) is an open domain QA dataset including about 43k trivia questions collected from various internet sources, and 100 supporting passages for each question. These supporting passages are given in an order ranked by a search engine.

WikiHop (Welbl et al., 2018) is a multiple choice QA dataset constructed using a structured knowledge base. One has to submit the model and work with the author to obtain the test score. For this dataset, a binary feature is concatenated with word embeddings and character embeddings to indicate whether a token is belong to any candidate answers.

The above three datasets have their official train/dev/test sets, so we do not split them by ourselves. Some of the above datasets provide additional meta-data, we do not use this additional information in our experiments. We observe that those low-ranked passages play a critical role in improving the accuracy, thus we remain all supporting passages as the inputs of our

Model	SearchQA		Quasar-T	
	EM	F1	EM	F1
DrQA	41.9	48.7	37.7	44.5
R^3	49.0	55.3	35.3	41.7
TraCRNet	52.9	65.1	43.2	54.0
Shared-Norm	59.8	67.1	38.6	45.4
HAS-QA	62.7	68.7	43.2	48.9
DynSAN	**64.2**	**70.3**	**48.0**	**54.8**
Human	43.9	–	51.5	60.6

Table 1: Performance of DynSAN and competing approaches on the test sets of two extractive QA tasks: SearchQA and Quasar-T. Competing approaches include DrQA (Chen et al., 2017), R^3 (Wang et al., 2018a), TraCRNet (Dehghani et al., 2019a), Shared-Norm (Clark and Gardner, 2018), HAS-QA (Pang et al., 2019). Human performance is referenced from the dataset paper.

model. The averages/medians of the total length of the concatenation of all supporting passages for each question are around 1.9k/2k, 2.4k/2.4k, and 1.2k/1k in SearchQA, Quasar-T, and WikiHop respectively. Thus, we limit the maximum length not to exceed 5k tokens and discard a few exceptionally long cases. Tokenization is completed using spaCy [1] during preprocessing.

4.2 Experimental Setup

In the DynSAN, the kernel size is 7 for all convolutional layers, the standard dimension D is 128, the number of heads H is 8, the number of chosen tokens K is 256. In the cross-passage attention layer, we stack $N = 4$ layers of DynSA blocks. The mini-batch size is set to 32. For regularization, we adopt dropout between every two layers and the dropout rate is 0.1. Adam (Kingma and Ba, 2015) with learning rate 0.001 is used for tuning the model parameters. We use a learning rate warm-up scheme in which the learning rate increases linearly from 0 to 0.001 in the first 500 steps. The models for multi-passage reading comprehension are trained on four 12GB K80 GPUs using synchronous SGD (Das et al., 2016). Exponential moving average is adopted with a decay rate 0.9999.

4.3 Main Results

The performance of our model and competing approaches are summarized in Table 1 and Table 2. For extractive QA, standard metrics are utilized:

[1] https://spacy.io

Model	Dev	Test
BiDAF (Seo et al., 2017)	–	42.9
Coref GRU (Dhingra et al., 2018)	56.0	59.3
MHQA-GRN (Song et al., 2018)	62.8	65.4
Entity-GCN (De Cao et al., 2019)	64.8	67.6
CFC (Zhong et al., 2019)	66.4	70.6
DynSAN	**70.1**	**71.4**
Human (Welbl et al., 2018)	–	74.1

Table 2: Performance of DynSAN and competing approaches on multiple choice QA dataset: WikiHop.

Exact Match (EM) and F1 score (Rajpurkar et al., 2016). The scores are evaluated by the official script in Rajpurkar et al. (2016). For multiple choice QA, the performance is evaluated by the accuracy of choosing the correct answer. As we can see, the proposed model clearly outperforms all previously published approaches and achieves new state-of-the-art performances on the three datasets, which validates the effectiveness of the dynamic self-attention network for multi-passage RC. It is noteworthy that competing approaches use coarse-grain representations for cross-passage information interaction or omit cross-passage information interaction entirely.

Ablation	EM	F1
Full architecture	64.2	70.3
(a) – Cross-passage attention	55.1	60.9
(b) – Self-attention	59.5	65.6
(c) – Convolutional layers	61.0	67.4
(d) – Gated token selection	60.5	66.6
(e) – Gate	59.9	66.0
(f) – Regularization ($\beta = 0$)	63.7	69.8
(g) Replace with Bi-BloSA	60.5	67.1
(h) + Convolutional layers	61.3	67.6

Table 3: Ablation study on SearchQA test set. "–"/"+" denotes removing/adding a model component, the indent in (e) and (h) means removing/adding a model component on the basis of the previous line.

4.4 Ablations

In order to evaluate the individual contribution of each model component, we conduct an ablation study. Explicitly, we remove or replace model components and report the performance on the SearchQA test set in Table 3. In (a), we remove the cross-passage attention. In (b), we remove all self-attention, i.e., the context information is modeled by the convolutional layers only. In (c), we

Question: Which vegetable is a Welsh emblem? Answer: leek Prediction: leek	Question: What gemstone was reputed to heal eye ailments? Answer: emerald Prediction: pearl
… A pungent vegetable is the national emblem of Wales …	... pearl was used therapeutically to heal eye ailments ...
… The leek (a vegetable) is a national emblem ...	... The gemstone gets its name from its resemblance to the eye of a tiger ...
… The vegetable called leek is also considered to …	... Copper is used by medical science for many ailments ...
... the reason why the daffodil is used as an emblem is ...	... Iris Agate: Use to heal burns ...
... Lochcarron of Scotland has a new Welsh Emblem ...	... 9th December 2008 Crystal Healing ...
... air force emblem ferrari prancing ...	

Figure 3: Case study on the Quasar-T dev set to show which tokens are selected as important tokens by the gated token selection mechanism in DynSA block. Important tokens are shaded.

remove all convolutional layers in DynSA blocks. In (d), we remove the gated token selection mechanism in DynSA blocks; in other words, which K tokens are selected is decided randomly rather than by the gate activation. Further, we remove the gate itself from (d) in (e). In (f), we remove regularization on gate activation by setting $\beta = 0$. In (g), we replace the DynSA block with Bi-BloSA (Shen et al., 2018b), which is proposed for long-sequence modeling but a block-level self-attention. The Bi-BloSA is implemented using the author's open source code. On the basis of (g), we combine Bi-BloSA with convolutional layers in (h).

As is shown in Table 3, cross-passage attention is most critical to the performance (almost 10% drop), the results prove the necessity of formation interaction between passages. Since we set $K = 256$, and most singular passages are within 256 tokens, the DynSA models local context for every position before the concatenation of all passages. Therefore, removing convolutional layers does not degrade the model entirely in (c). Self-attention and convolutional layers account for 4.7% and 2.9% performance drop respectively, and it illustrates that self-attention plays a more critical role than convolutional layers in modeling context information. In (d), the performance reduces significantly, proving the effectiveness of the gated token selection mechanism in the proposed architecture. Compare (e) to (d) and compare (f) to the full architecture, it is concluded that the gate itself and the regularization also have slight benefits to the model. From (g) and (h), we learn that the token-level DynSA block outperforms the block-level Bi-BloSA by a large margin, verifying the superiority of fine-grain representation.

4.5 Qualitative Analysis

We conduct a case study to show which tokens are selected as important tokens by the gated token selection mechanism. In a DynSA block, we define the maximum gate activation in all heads as a token's activity. The activity reflects the estimated importance of a token. In this subsection, all the tokens are ranked according to the sum of a token's activities in all DynSA blocks in the cross-passage attention layer. In Figure 3, two question-answering instances are given, and the top-ranked tokens are shaded. As we can see, the model inclines to mark cue words and plausible answers as the important tokens in DynSA blocks. We conjecture that information interactions between plausible answers may play an answer verification role,

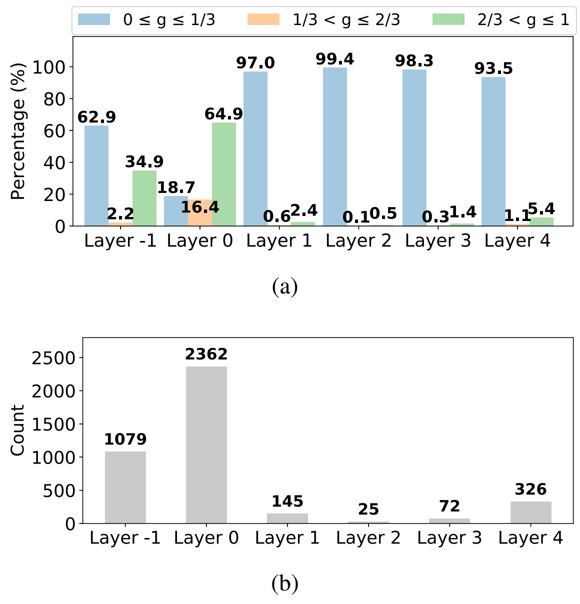


Figure 4: Quantitative analysis on the Quasar-T dev set. Layers are indexed from the bottom up, the DynSA blocks in the cross-passage attention layer are indexed from layer 1 to layer 4, the two DynSA blocks below the cross-passage attention layer are indexed as layer -1 and layer 0 respectively. (a) The distribution of the number of tokens of different activities in each layer. Tokens are classified into three categories according to its activity value g. (b) The average amount of active tokens ($g > 0.01$) in each layer.

SQuAD 1.1	Speedup	Memory	$\lvert\theta\rvert$	EM/F1
Bi-LSTM	1.0**x**/1.0**x**	4305	1.3M	70.5/79.8
Full SAN	3.5**x**/2.5**x**	8748	1.9M	70.6/80.1
Bi-BloSAN	3.4**x**/2.3**x**	6414	1.9M	66.7/76.8
DynSAN	4.3**x**/3.3**x**	4341	1.9M	69.9/79.5

Table 4: The time cost and memory consumption on SQuAD. The time cost is shown through the speedup rate with respect to Bi-LSTM. Both the training speedup rate and inference speedup rate are reported. The memory usage is measured in Megabyte. $|\theta|$ denotes the amount of trainable parameters in a model. Accuracy is measured by EM and F1.

while information interactions between cue words may be considered as multihop reasoning. We also observe that in a lot of mispredicted instances the correct answer never obtains large gate activations in cross-passage attention layers. Perhaps this is a reason for misprediction.

4.6 Quantitative Analysis

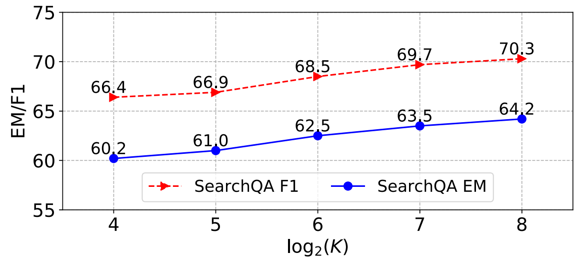
Figure 4(a) illustrates the distribution of the number of tokens of different activities in each layer. Token's activity is defined as in subsection 4.5. We also count the average number of active tokens on the Quasar-T dev set. We define a token is active when its activity is greater than 0.01. Figure 4(b) reports the statistics. In general, the activity values tend to be polarized, i.e., either near zero or near one. It is probably caused by the normalization in Equation 3 and the regularization term in Equation 6. Besides, the intra-passage DynSA blocks (layer -1 and layer 0) have more active tokens, while the cross-passage blocks have less. It explains that more tokens take effect in understanding a single passage, while only a few important tokens are necessary for cross-passage information interaction. The results verify our observation mentioned in section 1.

4.7 Time Cost & Memory Consumption

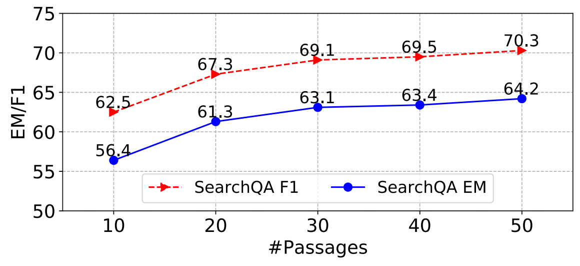
We also conduct experiments to show the computational costs of the proposed model and other baseline models. Specifically, we replace the DynSA blocks in Figure 2 with Bi-LSTM (Hochreiter and Schmidhuber, 1997), full SAN, and Bi-BloSAN (Shen et al., 2018b) respectively. Note that the full SAN refers to the model encoder block in QANet (Yu et al., 2018), which is a combination of global multi-head self-attention and local convolution. It is a strong baseline, and we use it to show the situation of full self-attention over all tokens.

To avoid the long running time of Bi-LSTM and the out-of-memory issue of full SAN on multi-passage RC tasks, we select SQuAD 1.1 (Rajpurkar et al., 2016) as the benchmark dataset. Since SQuAD is a single-passage RC task, we consider it as special multi-passage RC when the number of passages M equals to 1. In this experiment, top $K = 32$ tokens are chosen in DynSAN. Models are trained on a single 12GB K80 GPU.

The results are shown in Table 4. Compared with the full SAN and Bi-LSTM, DynSAN has a slight accuracy drop while Bi-BloSAN degrades significantly. In terms of time cost and memory usage, DynSAN reaches 4.3**x** and 3.3**x** speedup and has a similar memory consumption to Bi-LSTM. Because of the characteristics of Bi-LSTM and the full SAN, as the sequence length increases, the advantage of DynSAN in speed and memory consumption would be more significant. Although DynSAN has a small accuracy drop to the full SAN, it seems that DynSAN is a relatively balanced model concerning speed, memory, and accuracy.



(a)



(b)

Figure 5: (a) Effects of choosing different values of the hyper-parameter K in token selection. K is the number of chosen tokens, and is set to a power of 2. (b) Performance against the number of supporting passages.

4.8 Model Analysis

Effect of Token Selection Figure 5(a) shows the effects of the token selection. As the number of chosen tokens increases, performance improves as expected. When the number of chosen tokens is large enough, the gain becomes marginal. The choice of this hyper-parameter has an impact on the balance in speed, memory, and accuracy.

Number of Passages Figure 5(b) answers following research question “How would the performance change with respect to the number of passages?” As more supporting passages are taken into consideration, both F1 and EM performance of our model continuously increase. The results verify that those low-ranked passages play a critical role in answering the questions.

5 Related Works

As far as multi-passage reading comprehension be concerned, a lot of powerful deep learning approaches have been introduced to solve this problem. De Cao et al. (2019); Song et al. (2018) introduce graph convolutional network (GCN) and graph recurrent network (GRN) into this task. Dhingra et al. (2018) use co-reference annotations extracted from an external system to connect entity mentions for multihop reasoning. Zhong et al. (2019) propose an ensemble approach for coarse-grain and fine-grain co-attention networks. Pang et al. (2019) propose a hierarchical answer spans model to tackle the problem of multiple answer spans. Clark and Gardner (2018) uses a shared-normalization objective to produce accurate per-passage confidence scores and marginalize the probability of an answer candidate over all passages. While it outperforms most single-passage RC models by a large margin, it processes each passage independently omitting the multi-passage information interaction completely. In Wang et al. (2018b), cross-passage answer verification is definitely proposed, in which all the word embeddings in a passage are summed through attention mechanism to represent an answer candidate, and then each answer candidate attends to other candidates to collect supportive information. In Dehghani et al. (2019a), multihop reasoning is implemented by a Universal Transformer (Dehghani et al., 2019b) which is mainly based on Multi-head Self-attention (Vaswani et al., 2017) and a transition function.

Our work is concerned with Self-attention Network (SAN) (Vaswani et al., 2017; Shen et al., 2018a). For the first time, Vaswani et al. (2017) explore the possibilities of completely replacing the recurrent neural network with self-attention to model context dependencies. Some papers propose variants of self-attention mechanisms, such as Shen et al. (2018c); Hu et al. (2018); Shaw et al. (2018); Yang et al. (2019). Besides, Shen et al. (2018b) explore reducing the computational complexity of self-attention.

6 Conclusion

In this paper, we proposed a new Dynamic Self-attention (DynSA) architecture, which dynamically determinates what tokens are important for constructing intra-passage or cross-passage token-level semantic representations. The proposed approach has the advantages in remaining fine-grain semantic information meanwhile reaching a balance between time, memory and accuracy. We showed the effectiveness of the proposed method in handling multi-passage reading comprehension using three benchmark datasets including SearchQA, Quasar-T, and WikiHop. Experimental results showed state-of-the-art performance.

Acknowledgments

We thank the anonymous reviewers for their valuable comments, and Johannes Welbl for evaluating our model on the hidden WikiHop test dataset.

References

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. 2016. Layer normalization. *arXiv preprint arXiv:1607.06450*.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *3rd International Conference on Learning Representations, ICLR*.

Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. 2017. Reading Wikipedia to answer open-domain questions. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1870–1879, Vancouver, Canada. Association for Computational Linguistics.

Kyunghyun Cho, Bart van Merrienboer, Dzmitry Bahdanau, and Yoshua Bengio. 2014. On the properties of neural machine translation: Encoder–decoder approaches. In *Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation*, pages 103–111, Doha, Qatar. Association for Computational Linguistics.

François Chollet. 2017. Xception: Deep learning with depthwise separable convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1251–1258.

Christopher Clark and Matt Gardner. 2018. Simple and effective multi-paragraph reading comprehension. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Melbourne, Australia. Association for Computational Linguistics.

Dipankar Das, Sasikanth Avancha, Dheevatsa Mudigere, Karthikeyan Vaidynathan, Srinivas Sridharan, Dhiraj Kalamkar, Bharat Kaul, and Pradeep Dubey. 2016. Distributed deep learning using synchronous stochastic gradient descent. *arXiv preprint arXiv:1602.06709*.

Nicola De Cao, Wilker Aziz, and Ivan Titov. 2019. Question answering by reasoning across documents with graph convolutional networks. *Conference of the North American Chapter of the Association for Computational Linguistics*.

Mostafa Dehghani, Hosein Azarbonyad, Jaap Kamps, and Maarten de Rijke. 2019a. Learning to transform, combine, and reason in open-domain question answering. In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*, pages 681–689. ACM.

Mostafa Dehghani, Stephan Gouws, Oriol Vinyals, Jakob Uszkoreit, and Łukasz Kaiser. 2019b. Universal transformers. *International Conference on Learning Representations*.

Bhuwan Dhingra, Qiao Jin, Zhilin Yang, William Cohen, and Ruslan Salakhutdinov. 2018. Neural models for reasoning over multiple mentions using coreference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 42–48, New Orleans, Louisiana. Association for Computational Linguistics.

Bhuwan Dhingra, Kathryn Mazaitis, and William W Cohen. 2017. Quasar: Datasets for question answering by search and reading. *arXiv preprint arXiv:1707.03904*.

Matthew Dunn, Levent Sagun, Mike Higgins, V Ugur Guney, Volkan Cirik, and Kyunghyun Cho. 2017. Searchqa: A new q&a dataset augmented with context from a search engine. *arXiv preprint arXiv:1704.05179*.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Minghao Hu, Yuxing Peng, Zhen Huang, Xipeng Qiu, Furu Wei, and Ming Zhou. 2018. Reinforced mnemonic reader for machine reading comprehension. *27th International Joint Conference on Artificial Intelligence*.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1746–1751, Doha, Qatar. Association for Computational Linguistics.

Diederik P Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. *International Conference on Learning Representations*.

Tomas Mikolov, Edouard Grave, Piotr Bojanowski, Christian Puhrsch, and Armand Joulin. 2018. Advances in pre-training distributed word representations. In *Proceedings of the International Conference on Language Resources and Evaluation (LREC 2018)*.

Liang Pang, Yanyan Lan, Jiafeng Guo, Jun Xu, Lixin Su, and Xueqi Cheng. 2019. Has-qa: Hierarchical answer spans model for open-domain question answering. *AAAI Conference on Artificial Intelligence*.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas. Association for Computational Linguistics.

Minjoon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. 2017. Bidirectional attention flow for machine comprehension. *International Conference on Learning Representations*.

Peter Shaw, Jakob Uszkoreit, and Ashish Vaswani. 2018. Self-attention with relative position representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 464–468, New Orleans, Louisiana. Association for Computational Linguistics.

Tao Shen, Tianyi Zhou, Guodong Long, Jing Jiang, Shirui Pan, and Chengqi Zhang. 2018a. Disan: Directional self-attention network for rnn/cnn-free language understanding. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Tao Shen, Tianyi Zhou, Guodong Long, Jing Jiang, and Chengqi Zhang. 2018b. Bi-directional block self-attention for fast and memory-efficient sequence modeling. *International Conference on Learning Representations*.

Tao Shen, Tianyi Zhou, Guodong Long, Jing Jiang, and Chengqi Zhang. 2018c. Fast directional self-attention mechanism. *arXiv preprint arXiv:1805.00912*.

Linfeng Song, Zhiguo Wang, Mo Yu, Yue Zhang, Radu Florian, and Daniel Gildea. 2018. Exploring graph-structured passage representation for multi-hop reading comprehension with graph neural networks. *arXiv preprint arXiv:1809.02040*.

Rupesh Kumar Srivastava, Klaus Greff, and Jürgen Schmidhuber. 2015. Highway networks. *arXiv preprint arXiv:1505.00387*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008.

Shuohang Wang, Mo Yu, Xiaoxiao Guo, Zhiguo Wang, Tim Klinger, Wei Zhang, Shiyu Chang, Gerry Tesauro, Bowen Zhou, and Jing Jiang. 2018a. R 3: Reinforced ranker-reader for open-domain question answering. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Yizhong Wang, Kai Liu, Jing Liu, Wei He, Yajuan Lyu, Hua Wu, Sujian Li, and Haifeng Wang. 2018b. Multi-passage machine reading comprehension with cross-passage answer verification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 1: Long Papers*, pages 1918–1927.

Dirk Weissenborn, Georg Wiese, and Laura Seiffe. 2017. Making neural QA as simple as possible but not simpler. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*, pages 271–280, Vancouver, Canada. Association for Computational Linguistics.

Johannes Welbl, Pontus Stenetorp, and Sebastian Riedel. 2018. Constructing datasets for multi-hop reading comprehension across documents. *Transactions of the Association of Computational Linguistics*, 6:287–302.

Baosong Yang, Longyue Wang, Derek F Wong, Lidia S Chao, and Zhaopeng Tu. 2019. Convolutional self-attention network. *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.

Adams Wei Yu, David Dohan, Minh-Thang Luong, Rui Zhao, Kai Chen, Mohammad Norouzi, and Quoc V Le. 2018. Qanet: Combining local convolution with global self-attention for reading comprehension. *International Conference on Learning Representations*.

Victor Zhong, Caiming Xiong, Nitish Shirish Keskar, and Richard Socher. 2019. Coarse-grain fine-grain coattention network for multi-evidence question answering. *International Conference on Learning Representations*.