Stochastic Answer Networks for Natural Language Inference

Xiaodong Liu[†], **Kevin Duh**[‡] **and Jianfeng Gao**[†]
[†] Microsoft Research, Redmond, WA, USA
[‡] Johns Hopkins University, Baltimore, MD, USA
[†]{xiaodl,jfgao}@microsoft.com [‡]kevinduh@cs.jhu.edu

Abstract

We propose a stochastic answer network (SAN) to explore multi-step inference strategies in Natural Language Inference. Rather than directly predicting the results given the inputs, the model maintains a state and iteratively refines 精炼 its predictions. Our experiments show that SAN achieves the state-of-the-art results on three benchmarks: Stanford Natural Language Inference (SNLI) dataset, MultiGenre Natural Language Inference (MultiNLI) dataset and Quora Question Pairs dataset.

1 Motivation

NLI又称RTE

The natural language inference task, also known as *recognizing textual entailment* (RTE), is to infer the relation between a pair of sentences (e.g., *premise* and *hypothesis*). This task is challenging, since it requires a model to fully understand the sentence meaning, (i.e., lexical and compositional semantics). For instance, the following example from MultiNLI dataset (Williams et al., 2017) illustrates the need for a form of multi-step synthesis 综合 of information between ***premise***: *"If you need this book, it is probably too late unless you are about to take an SAT or GRE."*, and ***hypothesis***: *"It's never too late, unless you're about to take a test."* To predict the correct relation between these two sentences, the model needs to first infer that "*SAT* or *GRE*" is a "*test*", and then pick the correct relation, e.g., *contradiction*.

This kind of iterative process can be viewed as a form of multi-step inference. The majority of works on NLI use a *single step* inference. Inspirited by the recent success of multi-step inference on Machine Reading Comprehension (MRC) (Hill et al., 2016; Dhingra et al., 2016; Sordoni et al., 2016; Kumar et al., 2015; Shen et al., 2017; Liu et al., 2018), we explore the multi-step inference strategies on NLI and we show that such alternatives significantly boost the performance. In particular, we adapt the multi-step model in (Liu et al., 2018) to RTE: Rather than directly predicting the entailment given the inputs, our model maintains a state and iteratively refines its predictions. This achieves the state-of-the-art on SNLI, MultiNLI and Quora Question Pairs datasets.

2 Proposed Model: SAN

The natural language inference task as defined here involves a premise $P = \{p_0, p_1, ..., p_{m-1}\}$ of m words and a hypothesis $H = \{h_0, h_1, ..., h_{n-1}\}$ of n words, and aims to find a logic relationship R between P and H, which is one of labels in a close set[1]: *entailment*, *neutral* and *contradiction*. The learning algorithm for NLI is to learn a function $f(P, H) \to R$.

In a single-step inference architecture, the model directly predicts R given P and H as input. In our multi-step inference architecture, we additionally incorporate a recurrent state s_t; the model processes multiple passes through P and H, iteratively refining the state s_t, before finally generating the output at step $t = T$, where T is an a priori chosen limit on the number of inference steps.

整体 SAN 模型的思想.

Figure 1 describes in detail the architecture of the **s**tochastic **a**nswer **n**etwork (SAN) we propose in this study. It contains four different layers:

Lexicon Encoding Layer. Our lexicon embedding is concatenation of word embeddings (300维), pre-trained 300-dimensional GloVe embedding (Pennington et al., 2014), and its character embedding (300维), a concatenation of the multi-filters Convolutional Neural Network [2] (MF-CNN) outputs to handle

MF-CNN 训练得到, 为了处理OOV的问题

[1] In SNLI, the label "-" denotes inconsistency between annotators, and it is removed during the training and test.

[2] In this paper, we use three different filters $5, 10, 15$ with the hidden size $50, 100, 150$, respectively. Thus, each to-

SAN 架构（四层）

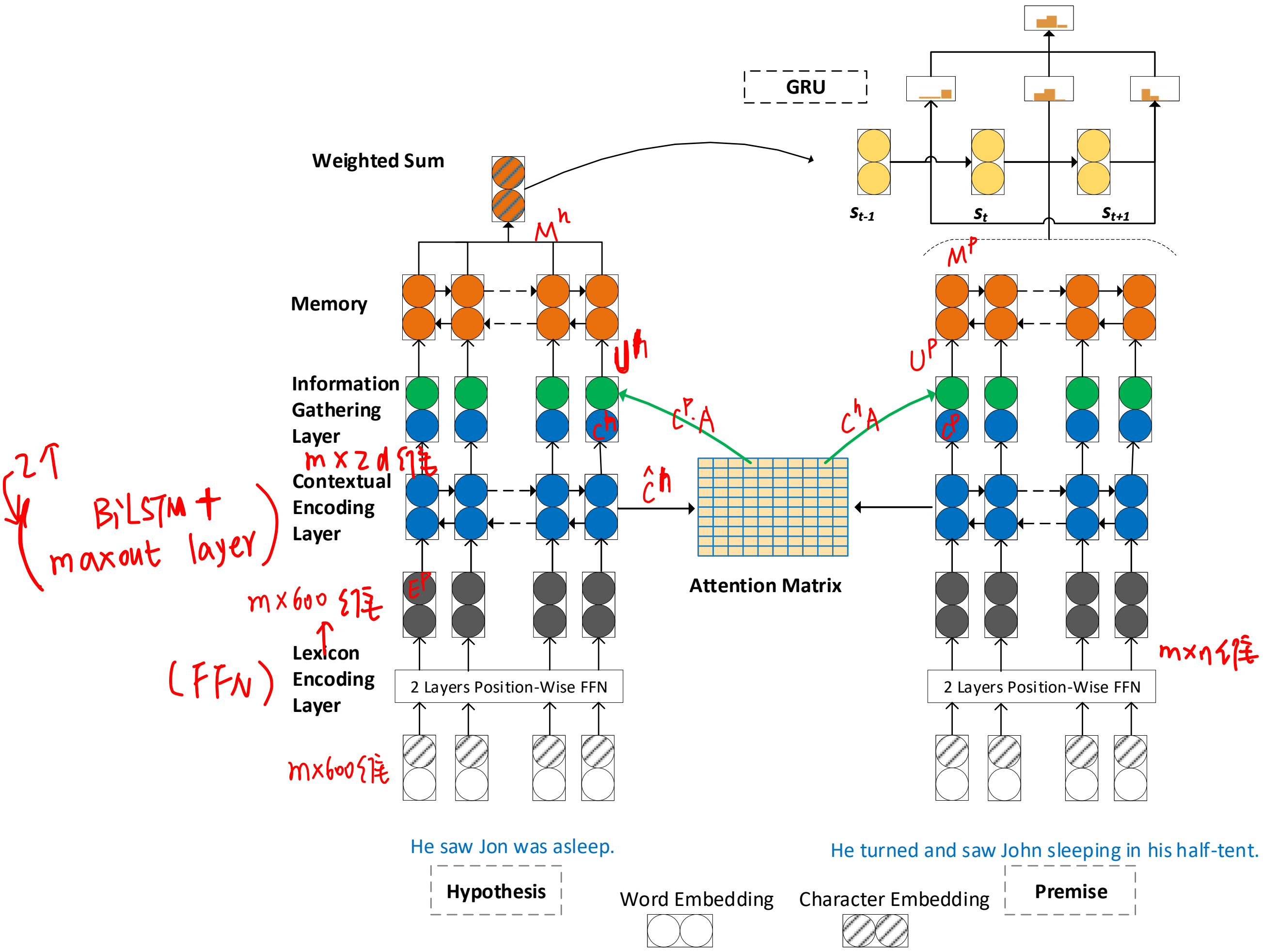

Figure 1: Architecture of the Stochastic Answer Network (SAN) for Natural Language Inference.

the out-of-vocabulary (OOV) issue. Inspired by (Vaswani et al., 2017), we use two separate two-layer position-wise FFN, defined as: $FFN(x) = W_2ReLU(W_1x + b_1) + b_2$, to project lexical encoding of both premise and hypothesis, and obtain the final lexicon embedings, $E^p \in \mathbb{R}^{d\times m}$ and $E^h \in \mathbb{R}^{d\times n}$, for the tokens in P and H , respectively. Here, d is the hidden size (the same as the hidden size of contextual embedding).

Contextual Encoding Layer. Two stacked BiLSTM layers are used on the lexicon encoding layer to encode the context information for each word in both P and H. Due to the bidirectional layer, it doubles the hidden size. We use a maxout layer (Goodfellow et al., 2013) on the BiLSTM to shrink its output into its original hidden size. By a concatenation of the outputs of two BiLSTM layers, we obtain $C^p \in \mathbb{R}^{2d\times m}$ and $C^h \in \mathbb{R}^{2d\times n}$ as representation of P and H, respectively.

Memory Layer. We construct our working memory via an attention mechanism. First, we project both C^p and C^h with one layer neural network $ReLU(W_3x)$ to obtain $\hat{C}^p$ and $\hat{C}^h$ (Liu et al., 2018), respectively. Then an attention matrix by using a dot-product attention (Vaswani et al., 2017) is computed:

$$A = dropout(f_{attention}(\hat{C}^p, \hat{C}^h)) \in \mathbb{R}^{m\times n} \quad (1)$$

Next, we gather all the information on premise and hypothesis by:

$$U^p = concat(C^p, C^hA) \in \mathbb{R}^{4d\times m} \quad (2)$$

$$U^h = concat(C^h, C^pA') \in \mathbb{R}^{4d\times n} \quad (3)$$

Here, A' is the transpose of A. At last, the working memory of the premise and hypothesis is generated by using a BiLSTM based on all the information gathered:

ken in the premise and hypothesis is represented as a 600-dimensional vector.

$$M^p = BiLSTM([U^p; C^p]); \quad (4)$$

$$M^h = BiLSTM([U^h; C^h]) \quad (5)$$

where the semicolon mark ; indicates the vector/matrix concatenation operator.

Answer module. Formally, our answer module will compute over T memory steps and output the relation label. At the beginning, the initial state s_0 is the summary of the M^h: $s_0 = \sum_j \alpha_j M_j^h$, where $\alpha_j = \frac{exp(w_4 \cdot M_j^h)}{\sum_{j'} exp(w_4 \cdot M_{j'}^h)}$. At time step t in the range of $\{1, 2, ..., T-1\}$, the state is defined by $s_t = GRU(s_{t-1}, x_t)$. Here, x_t is computed from the previous state s_{t-1} and memory M^p: $x_t = \sum_j \beta_j M_j^p$ and $\beta_j = softmax(s_{t-1} W_5 M^p)$. Following (Mou et al., 2015), one layer classifier is used to determine the relation at each step $t \in \{0, 1, \ldots, T-1\}$.

$$P_t^r = softmax(W_6[s_t; x_t; |s_t - x_t|; s_t \cdot x_t]). \quad (6)$$

At last, we utilize all of the T outputs by averaging the scores:

$$P^r = avg([P_0^r, P_1^r, ..., P_{T-1}^r]). \quad (7)$$

最终的预测结果

Each P_t^r is a probability distribution over all the relations, $\{1, \ldots, |R|\}$. During training, we apply *stochastic prediction dropout* before the above averaging operation. During decoding, we average all outputs to improve robustness.

This stochastic prediction dropout is similar in motivation to the dropout introduced by (Srivastava et al., 2014). The difference is that theirs is dropout at the intermediate node-level, whereas ours is dropout at the final layer-level. Dropout at the node-level prevents correlation between features. Dropout at the final layer level, where randomness is introduced to the averaging of predictions, prevents our model from relying exclusively on a particular step to generate correct output.

3 Experiments

3.1 Dataset

MultiNLI
SNLI
Quora Question Pairs datasets

Here, we evaluate our model on three benchmark datasets (MultiNLI, SNLI and Quora Question Paris datasets), and the evaluation metric for all the datasets is accuracy. 指标 **SNLI** (Bowman et al., 2015) contains 570k human annotated sentence pairs, in which the premises are drawn from the captions of the Flickr30 corpus, and hypothesis are manually annotated. **MultiNLI** (Williams et al., 2017) contains 433k sentence pairs, which are collected similarly as SNLI. However, the premises are collected from a broad range of genre of American English. The test and development sets are further divided into in-domain (**matched**) and cross-domain (**mismatched**) sets. The Quora Question Pairs dataset (Wang et al., 2017) is proposed for paraphrase identification. It contains 400k question pairs, and each question pair is annotated with a binary value indicating whether the two questions are paraphrase of each other.

	Single-step	SAN
MultiNLI matched	78.69	**79.88**
MultiNLI mismatched	78.83	**79.91**
SNLI	88.32	**88.73**
Quora	89.67	**89.70**

Table 1: Comparison of single and multi-step inference strategies on MultiNLI, SNLI and Quora Question **dev** sets.

3.2 Implementation details

分词工具

The spaCy tool[3] is used to tokenize all the dataset and PyTorch[4] is used to implement our models. For the character encoding, we limit the maximum length of a word by 20 characters and padding a special *padding* token if it is shorter than 20. The character embedding size is set to 20. We fix word embedding with 300-dimensional GloVe word vectors (Pennington et al., 2014). The embedding for the out-of-vocabulary (OOV) is zeroed. The hidden size of LSTM in the contextual encoding layer, memory generation layer is set to 128, thus the input size of output layer is 1024 (128 * 2 * 4) as Eq 6. The projection size in the attention layer is set to 256. To speed up training, we use weight normalization (Salimans and Kingma, 2016). The dropout rate is 0.2, and the dropout mask is fixed through time steps (Gal and Ghahramani, 2016) in LSTM. The mini-batch size is set to 32. Our optimizer is Adamax (Kingma and Ba, 2014) and its learning rate is initialized as 0.002 and decreased by 0.5 after each 10 epochs.

word长度是20个字符

超参设置

[3] https://spacy.io
[4] http://pytorch.org

3.3 Results

One main question which we would like to address is whether the multi-step inference help on NLI. We fixed the lower layer and only compare different architectures for the output layer:

1. *Single-step*: Predict the relation using Eq 6 based on s_0 and x_0. Here, $x_0 = \sum_j \alpha_j M_j^p$, where $\alpha_j = \frac{exp(w \cdot M_j^p)}{\sum_{j'} exp(w \cdot M_{j'}^p)}$ [5].

2. *SAN*: The proposed multi-step inference model. Note that we use 5-steps with the prediction dropout rate 0.2 on the all experiments.

Table 1 shows that our multi-step model outperforms the single-step model on the dev set of all three datasets. For example, on MultNLI matched condition, SAN achieved 79.88 while single-step achieves 78.69.

We compare our results with the state-of-the-art in Table 2. Our model achieves the best performance on these tasks. Due to the space limitation, we only list two top models (ref: MultiNLI[6,7], SNLI[8] and Quora Question Pairs (Gong et al., 2017) for more information.)

Analysis We analyze our model on the annotated subset[9] of development set of MultiNLI provided by (Williams et al., 2017). It contains 1,000 examples and each sample is tagged with zero or more labels, in total 13 categories, as shown in Table 3. We observe that our model outperforms the best system in RepEval 2017 in most cases, except on "Conditional" and "Tense Difference" categories. In particular, on the most challenging "Long Sentence" and "Quantity/Time" categories, SAN's result is substantially better than previous systems'. This demonstrates the robustness of our model.

Model	MultiNLI Test	
	Matched	Mismatched
DIIN(Gong et al., 2017)	78.8	77.8
CAFE(Tay et al., 2018)	78.7	77.9
SAN	**79.3**	**78.7**
Ensemble	Matched	Mismatched
DIIN(Gong et al., 2017)	80.0	78.7
CAFE(Tay et al., 2018)	80.2	79.0
SAN*	**80.6**	**80.1**
SNLI Dataset (Accuracy%)		
CAFE(Tay et al., 2018)	88.5	
KIM(Chen et al., 2017)	**88.6**	
SAN	88.5	
Quora Question Dataset (Accuracy%)		
DecAtt(Tomar et al., 2017)	88.4	
DIIN(Gong et al., 2017)	89.1	
SAN	**89.4**	

Table 2: Comparison with the state-of-the-art on MultiNLI, SNLI and Quora Question **test** sets. *We ensemble three models trained with three different initialization seeds.

Tag	Matched		Mismatched	
	Chen[1]	SAN	Chen[1]	SAN
Conditional	**100%**	65%	**100%**	81%
Word overlap	63 %	**86%**	76%	**92%**
Negation	75%	**80%**	72%	**79%**
Antonym	50%	**77%**	58%	**85%**
Long Sentence	67%	**84%**	67%	**79%**
Tense Difference	**86%**	75%	**89%**	83%
Active/Passive	88%	**100%**	91%	**100%**
Paraphrase	78%	**92%**	89%	**92%**
Quantity/Time	33%	**53%**	46%	**51%**
Coreference	**83%**	73%	80%	**84%**
Quantifier	74%	**81%**	77%	**80%**
Modal	75%	**79%**	76%	**82%**
Belief	73%	**77%**	74%	**78%**

Table 3: Error analysis on MultiNLI dataset. [1]Refer (Nangia et al., 2017) for more details.

4 Conclusion

We explored the use of multi-step inference in natural language inference by proposing a stochastic answer network (SAN). Rather than directly predicting the results (e.g. relation R such as entailment or not) given the input premise P and hypothesis H, SAN maintains a state s_t, which it iteratively refines over multiple passes on P and H in order to make a prediction. Our state-of-the-art results on three benchmarks (SNLI, MultiNLI, Quora Question Pairs) show the effectiveness of

[5]For a fair comparison, we fix the lower layer and only change the output layer to the popular one as (Mou et al., 2015).

[6]https://www.kaggle.com/c/multinli-matched-open-evaluation

[7]https//www.kaggle.com/c/multinli-mismatched-open-evaluation

[8]https://nlp.stanford.edu/projects/snli

[9]https://www.nyu.edu/projects/bowman/multinli/multinli_1.0_annotations.zip

this multi-step inference architecture.

References

Samuel R Bowman, Gabor Angeli, Christopher Potts, and Christopher D Manning. 2015. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 632–642.

Q. Chen, X. Zhu, Z.-H. Ling, D. Inkpen, and S. Wei. 2017. Natural Language Inference with External Knowledge. *ArXiv e-prints*.

Bhuwan Dhingra, Hanxiao Liu, William W Cohen, and Ruslan Salakhutdinov. 2016. Gated-attention readers for text comprehension. *arXiv preprint arXiv:1606.01549*.

Yarin Gal and Zoubin Ghahramani. 2016. A theoretically grounded application of dropout in recurrent neural networks. In *Advances in neural information processing systems*, pages 1019–1027.

Y. Gong, H. Luo, and J. Zhang. 2017. Natural Language Inference over Interaction Space. *ArXiv e-prints*.

Ian J Goodfellow, David Warde-Farley, Mehdi Mirza, Aaron Courville, and Yoshua Bengio. 2013. Maxout networks. *arXiv preprint arXiv:1302.4389*.

Felix Hill, Antoine Bordes, Sumit Chopra, and Jason Weston. 2016. The goldilocks principle: Reading children's books with explicit memory representations. *ICLR*.

Diederik Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Ankit Kumar, Ozan Irsoy, Jonathan Su, James Bradbury, Robert English, Brian Pierce, Peter Ondruska, Ishaan Gulrajani, and Richard Socher. 2015. Ask me anything: Dynamic memory networks for natural language processing. *CoRR*, abs/1506.07285.

Xiaodong Liu, Yelong Shen, Kevin Duh, and Jianfeng Gao. 2018. Stochastic answer networks for machine reading comprehension. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics.

Lili Mou, Rui Men, Ge Li, Yan Xu, Lu Zhang, Rui Yan, and Zhi Jin. 2015. Natural language inference by tree-based convolution and heuristic matching. *arXiv preprint arXiv:1512.08422*.

N. Nangia, A. Williams, A. Lazaridou, and S. R. Bowman. 2017. The RepEval 2017 Shared Task: Multi-Genre Natural Language Inference with Sentence Representations. *ArXiv e-prints*.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Tim Salimans and Diederik P Kingma. 2016. Weight normalization: A simple reparameterization to accelerate training of deep neural networks. In *Advances in Neural Information Processing Systems*, pages 901–909.

Yelong Shen, Xiaodong Liu, Kevin Duh, and Jianfeng Gao. 2017. An empirical analysis of multiple-turn reasoning strategies in reading comprehension tasks. *arXiv preprint arXiv:1711.03230*.

Alessandro Sordoni, Philip Bachman, Adam Trischler, and Yoshua Bengio. 2016. Iterative alternating neural attention for machine reading. *arXiv preprint arXiv:1606.02245*.

Nitish Srivastava, Geoffrey E Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(1):1929–1958.

Y. Tay, L. A. Tuan, and S. Cheung Hui. 2018. A Compare-Propagate Architecture with Alignment Factorization for Natural Language Inference. *ArXiv e-prints*.

Gaurav Singh Tomar, Thyago Duque, Oscar Täckström, Jakob Uszkoreit, and Dipanjan Das. 2017. Neural paraphrase identification of questions with noisy pretraining. *arXiv preprint arXiv:1704.04565*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *arXiv preprint arXiv:1706.03762*.

Zhiguo Wang, Wael Hamza, and Radu Florian. 2017. Bilateral multi-perspective matching for natural language sentences. *arXiv preprint arXiv:1702.03814*.

A. Williams, N. Nangia, and S. R. Bowman. 2017. A Broad-Coverage Challenge Corpus for Sentence Understanding through Inference. *ArXiv e-prints*.