Controlling the Transition of Hidden States for Neural Machine Translation

Zaixiang Zheng Shujian Huang* Xin-Yu Dai Jiajun Chen

{zhengzx, huangsj, dxy, chenjj}@nlp.nju.edu.cn
Nanjing University

Oct 26, 2018

Outline

Outline

Neural Machine Translation

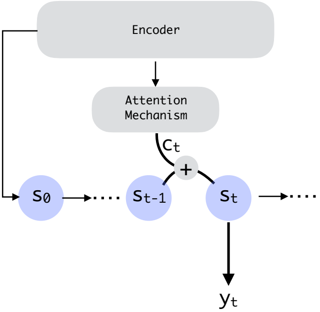
- Neural network-based methods show a promising trend in machine translation
- Generally, a neural machine translation (NMT) system adopts an *encoder-decoder* architecture with *attention mechanism* to model translating process

Recurrent Neural Network (RNN) based NMT

RNN-based NMT models with attention are widely deployed that deserves further study.

- Bi-directional encoder
- Uni-directional (left-to-right) decoder
- Encoder and decoder are bridged by attention

In this work, we focus on the RNN-based NMT model, especially its decoder part.



Recurrent Neural Network (RNN) based NMT

The NMT model

- is fed with a source sentence: $\langle x_1, ..x_i, ...\rangle \rightarrow \langle \boldsymbol{h}_1, ..\boldsymbol{h}_i, ...\rangle$
- does some **magics** with decoder **hidden states** step by step

$$\boldsymbol{s}_t = f(\boldsymbol{s}_{t-1}, y_{t-1}), \tag{1}$$

$$\boldsymbol{s}_0 = \text{summary}(\mathbf{h}). \tag{2}$$

- finally throws out a translated sentence: $\mathbf{s} \rightarrow \langle y_1, ..y_t, ...\rangle$

$$y_t = \text{softmax}(g(\boldsymbol{s}_t)). \tag{3}$$

Recurrent Neural Network (RNN) based NMT

The NMT model

- is fed with a source sentence: $\langle x_1, ..x_i, ...\rangle \rightarrow \langle \boldsymbol{h}_1, ..\boldsymbol{h}_i, ...\rangle$
- does some **magics** with decoder **hidden states** step by step

$$\boldsymbol{s}_t = f(\boldsymbol{s}_{t-1}, y_{t-1}), \tag{1}$$

$$\boldsymbol{s}_0 = \text{summary}(\mathbf{h}). \tag{2}$$

- finally throws out a translated sentence: $\mathbf{s} \rightarrow \langle y_1, ..y_t, ...\rangle$

$$y_t = \text{softmax}(g(\boldsymbol{s}_t)). \tag{3}$$

We may ask..

What happens inside this so-called "black box"?
Why and how does it work?

The Contents of The Hidden States

- If we could peep into the contents of the hidden states, we would get some inspirations

The Contents of The Hidden States

- If we could peep into the contents of the hidden states, we would get some inspirations
  - The contents of the hidden representations are predictive of several surface, syntactic and semantic attributes (sentence length, tense, etc) (Conneau et al., 2018)

The Contents of The Hidden States

- If we could peep into the contents of the hidden states, we would get some inspirations
  - The contents of the hidden representations are predictive of several surface, syntactic and semantic attributes (sentence length, tense, etc) (Conneau et al., 2018)
  - In each decoding step, the decoder hidden state is able to predict the rest untranslated Bag-of-Words (Weng et al., 2017)

The Contents of The Hidden States

- If we could peep into the contents of the hidden states, we would get some inspirations
  - The contents of the hidden representations are predictive of several surface, syntactic and semantic attributes (sentence length, tense, etc) (Conneau et al., 2018)
  - In each decoding step, the decoder hidden state is able to predict the rest untranslated Bag-of-Words (Weng et al., 2017)
  - The decoder hidden states should model past, present and future translation contents, which are varied according to the translation process (Zheng et al., 2018)

Preliminary Probing Experiment

We first conducted a preliminary probing experiment to explore the contents of the hidden states. We applied Bag-of-Word (BoW) predictions (Weng et al., 2017) on decoder hidden states.

Preliminary Probing Experiment

We first conducted a preliminary probing experiment to explore the contents of the hidden states. We applied Bag-of-Word (BoW) predictions (Weng et al., 2017) on decoder hidden states.

- we first trained an RNN-based NMT model

Preliminary Probing Experiment

We first conducted a preliminary probing experiment to explore the contents of the hidden states. We applied Bag-of-Word (BoW) predictions (Weng et al., 2017) on decoder hidden states.

- we first trained an RNN-based NMT model
- we built two word predictors (Weng et al., 2017) on the top of each decoder hidden states of the trained model to predict the BoW of forward and backward directions, respectively

Preliminary Probing Experiment

We first conducted a preliminary probing experiment to explore the contents of the hidden states. We applied Bag-of-Word (BoW) predictions (Weng et al., 2017) on decoder hidden states.

- we first trained an RNN-based NMT model
- we built two word predictors (Weng et al., 2017) on the top of each decoder hidden states of the trained model to predict the BoW of forward and backward directions, respectively
- we trained the word predictors while fixed all the parameters of the original NMT model

Preliminary Probing Experiment

We first conducted a preliminary probing experiment to explore the contents of the hidden states. We applied Bag-of-Word (BoW) predictions (Weng et al., 2017) on decoder hidden states.

- we first trained an RNN-based NMT model
- we built two word predictors (Weng et al., 2017) on the top of each decoder hidden states of the trained model to predict the BoW of forward and backward directions, respectively
- we trained the word predictors while fixed all the parameters of the original NMT model
- we wanted to validate if the hidden states store the translated and untranslated BoW, extending the observations of Weng et al. (2017)

Bag-of-Word (BoW) Predictions

$$acc_{fw} = \frac{1}{T}\sum_{1}^{T}\sum_{w\in top_{(\text{T-t-1})}(P_t)} \frac{\mathbb{1}(w \in y_{>(t+1)})}{T-t-1}$$

	Acc.	ppl
Forward pred.	71%	5.1
Backward pred.	78%	4.6

Table 1: Statistics of BoW predictions.

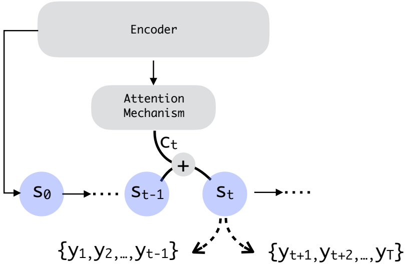


Figure 1: Illustration of BoW pred.

Bag-of-Word (BoW) Predictions

$$acc_{fw} = \frac{1}{T}\sum_{1}^{T}\sum_{w \in top_{(\text{T-t-1})}(P_t)} \frac{1(w \in y_{>(t+1)})}{T-t-1}$$

	Acc.	ppl
Forward pred.	71%	5.1
Backward pred.	78%	4.6

Table 1: Statistics of BoW predictions.

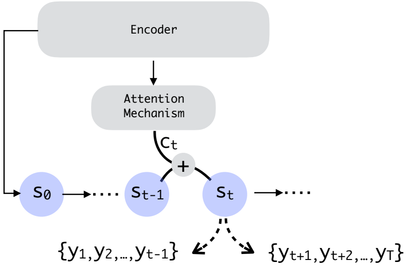


Figure 1: Illustration of BoW pred.

Each hidden state stores its backward directional contents (translated words) and forward directional contents (untranslated words).

Transitions of the Hidden States Matter

- Each hidden state stores *span-level* information of the complete translations.
- The changed factor is the transitions of the hidden states.

Transitions of the Hidden States Matter

- Each hidden state stores *span-level* information of the complete translations.
- The changed factor is the transitions of the hidden states.

We suggest that

- The translations vary at different timesteps in accordance with the hidden states' transitions.

Transitions of the Hidden States Matter

- Each hidden state stores *span-level* information of the complete translations.
- The changed factor is the transitions of the hidden states.

We suggest that

- The translations vary at different timesteps in accordance with the hidden states' transitions.
- The transitions play an important role in the RNN-based decoder, updating the span-level information of each hidden state.

Transitions of the Hidden States Matter

- Each hidden state stores *span-level* information of the complete translations.
- The changed factor is the transitions of the hidden states.

We suggest that

- The translations vary at different timesteps in accordance with the hidden states' transitions.
- The transitions play an important role in the RNN-based decoder, updating the span-level information of each hidden state.
- The difference between two hidden states should represent *lexicon-level* information of the current translation.

What Can We Do with It?

- The transitions of the hidden states play a important role in RNN-based decoder
- Regular MLE training does not guide the transition directly

What Can We Do with It?

- The transitions of the hidden states play a important role in RNN-based decoder
- Regular MLE training does not guide the transition directly

So, we probably need

An explicit supervision to control the transition

Outline

Predictive Constraint

For clarity, let us take $\{\boldsymbol{s}_{t-1}, \boldsymbol{s}_t\}$ and y_t for example.

- we denote $\boldsymbol{\Delta s}_t$ as the increment produced by the transition from $\boldsymbol{s}_{t-1}$ to $\boldsymbol{s}_t$.

Predictive Constraint

For clarity, let us take $\{\boldsymbol{s}_{t-1}, \boldsymbol{s}_t\}$ and y_t for example.

- we denote $\boldsymbol{\Delta s}_t$ as the increment produced by the transition from $\boldsymbol{s}_{t-1}$ to $\boldsymbol{s}_t$.
- the transition of two successive decoder hidden states should be predictive of the translation at current timestep.
  - $\boldsymbol{\Delta s}_t \approx y_t$,

Predictive Constraint

For clarity, let us take $\{\boldsymbol{s}_{t-1}, \boldsymbol{s}_t\}$ and y_t for example.

- we denote $\boldsymbol{\Delta s}_t$ as the increment produced by the transition from $\boldsymbol{s}_{t-1}$ to $\boldsymbol{s}_t$.
- the transition of two successive decoder hidden states should be predictive of the translation at current timestep.
  - $\boldsymbol{\Delta s}_t \approx y_t$,
  - we introduce a predictive constraint:

$$q(y_t|\boldsymbol{\Delta s}_t) = \mathrm{softmax}\big(\boldsymbol{E}(y_t)^\top \boldsymbol{W} \tanh(\boldsymbol{\Delta s}_t)\big), \tag{4}$$

where $\boldsymbol{W}$ is a learned matrix.

Predictive Constraint

$$q(y_t|\boldsymbol{\Delta s}_t) = \mathrm{softmax}\big(\boldsymbol{E}(y_t)^\top \boldsymbol{W} \tanh(\boldsymbol{\Delta s}_t)\big)$$

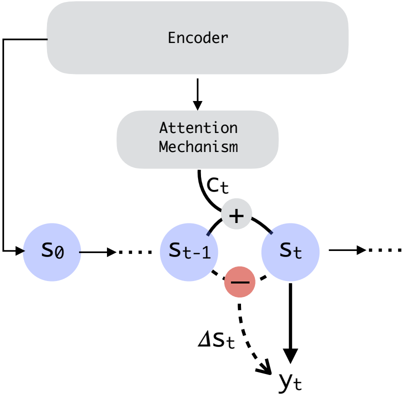


Figure 2: Illustration of proposed approach. Dotted lines denote the way to obtain the increment and predictive constraint.

How to Model the Transition Δs_t?

- Algebraic Subtraction
  - A general assumption: the decoding states form a shared latent representation space,
  - Get $\boldsymbol{\Delta s}_t$ by an algebraic subtraction

$$\boldsymbol{\Delta s}_t = \boldsymbol{s}_t - \boldsymbol{s}_{t-1} \tag{5}$$

How to Model the Transition Δs_t?

- Algebraic Subtraction
  - A general assumption: the decoding states form a shared latent representation space,
  - Get $\boldsymbol{\Delta s}_t$ by an algebraic subtraction

$$\boldsymbol{\Delta s}_t = \boldsymbol{s}_t - \boldsymbol{s}_{t-1} \tag{5}$$

- Parametric Subtraction
  - Apply the subtraction in a parametric manner
  - The subtrahend and minuend are first mapped by separate linear transformations $\boldsymbol{U}_1$ and $\boldsymbol{U}_2$, respectively

$$\boldsymbol{\Delta s}_t = \boldsymbol{U}_1 \boldsymbol{s}_t - \boldsymbol{U}_2 \boldsymbol{s}_{t-1} \tag{6}$$

Training

Give a training dataset $\{[\mathbf{x^{(m)}}, \mathbf{y^{(m)}}]\}_{m=1}^{M}$, we learn our model as follow:

- Sentence-level predictive constraint of the transitions

$$q_\gamma(\mathbf{y}) = \sum_{t}^{T} q(y_t | \boldsymbol{\Delta s}_t)$$

- Training objective

$$\mathcal{L}(\theta, \gamma) = \frac{1}{M} \sum_{m=1}^{M} \log P_\theta(\mathbf{y^{(m)}} | \mathbf{x^{(m)}}) + \log q_\gamma(\mathbf{y}^{(m)}) \tag{7}$$

Inference

- $\boldsymbol{\Delta s}_t$ is supposed to predict y_t,
- Use $P(y_t|y_{<t}) + q(y_t|\boldsymbol{\Delta s}_t)$ instead of $P(y_t|y_{<t})$ as the search score in testing phase

Inference

- $\boldsymbol{\Delta s}_t$ is supposed to predict y_t,
- Use $P(y_t|y_{<t}) + q(y_t|\boldsymbol{\Delta s}_t)$ instead of $P(y_t|y_{<t})$ as the search score in testing phase

This *re-scoring* strategy ensures the consistence of the objective of training and inference.

Outline

Settings

- Dataset
  - Chinese$\rightarrow$ English (Zh-En): NIST corpus, 1.6m
  - German $\leftrightarrow$ English (De-En & En-De): WMT2017 news translation task, 5.8m
- BPE for De-En and En-De,
- 30K vocabularies for Zh-En,
- Filter long sentence whose lengths are large than 80,
- 512-dims word embedding, 1024-dims hidden state,
- Use Adam with learning rate annealing for optimization.

Results on Zh-En

Model	MT03	MT04	MT05	Avg.	Δ
RNNSearch	37.95	40.80	36.06	38.27	-
Algebraic Subtraction	38.66	40.93	37.00	38.86	+0.59
Parametric Subtraction	39.07	41.23	37.35	39.22	+0.95
Algebraic Subtraction + Re-scoring	39.53	41.88	37.40	39.60	+1.33
Parametric Subtraction + Re-scoring	39.95	42.53	38.17	40.22	+1.95

Results on Zh-En

Model	MT03	MT04	MT05	Avg.	Δ
RNNSearch	37.95	40.80	36.06	38.27	-
Algebraic Subtraction	38.66	40.93	37.00	38.86	+0.59
Parametric Subtraction	39.07	41.23	37.35	39.22	+0.95
Algebraic Subtraction + Re-scoring	39.53	41.88	37.40	39.60	+1.33
Parametric Subtraction + Re-scoring	39.95	42.53	38.17	40.22	+1.95
RNNSearch (BPE 32K)	40.59	41.65	37.73	40.00	-
Parametric Subtraction + Re-scoring	41.43	43.50	39.83	41.59	+1.60

Table 2: Case-insensitive BLEU on Zh-En translation task.

Results on Zh-En

Model	MT03	MT04	MT05	Avg.	Δ
RNNSearch	37.95	40.80	36.06	38.27	-
Algebraic Subtraction	38.66	40.93	37.00	38.86	+0.59
Parametric Subtraction	39.07	41.23	37.35	39.22	+0.95
Algebraic Subtraction + Re-scoring	39.53	41.88	37.40	39.60	+1.33
Parametric Subtraction + Re-scoring	39.95	42.53	38.17	40.22	+1.95
RNNSearch (BPE 32K)	40.59	41.65	37.73	40.00	-
Parametric Subtraction + Re-scoring	41.43	43.50	39.83	41.59	+1.60

Table 2: Case-insensitive BLEU on Zh-En translation task.

Observations:

- Constraint on the transitions is effective
- Parametric constraint is better
- Re-scoring is useful and cheap to boost the performance

Results on De-En and En-De

Model	**De-En**		**En-De**	
	Dev	Test	Dev	Test
RNNSearch	32.0	27.8	28.3	23.3
Parametric Subtraction	32.2	28.7	29.6	23.6
Parametric Subtraction + re-scoring	32.9	29.1	30.6	24.1

Table 3: Case-sensitive BLEU on De-En and En-De Translation Tasks.

Results on De-En and En-De

Model	De-En		En-De	
	Dev	Test	Dev	Test
RNNSearch	32.0	27.8	28.3	23.3
Parametric Subtraction	32.2	28.7	29.6	23.6
Parametric Subtraction + re-scoring	32.9	29.1	30.6	24.1

Table 3: Case-sensitive BLEU on De-En and En-De Translation Tasks.

Observations:

- Our approach is effective across various language pairs,
- Our approach works well consistently on both words and sub-words (BPE) scenarios.

Analysis on Parameters and Speeds

Model	#Parameter	Speed	
		Training	Testing
RNNSearch	80M	42.59	2.05
Algebraic Subtraction	80.5M	37.91	2.03
Parametric Subtraction	82.5M	36.50	2.00
Parametric Subtraction + re-ordering	82.5M	36.55	1.72

Table 4: Statistics of parameters, training and testing speeds (sentences per second). **Note that** if we don't use re-scoring strategy, the newly added parameters will never be used in testing phase. i.e., it uses the same amount of parameters as the original NMT.

Analysis on Parameters and Speeds

Model	#Parameter	Speed	
		Training	Testing
RNNSearch	80M	42.59	2.05
Algebraic Subtraction	80.5M	37.91	2.03
Parametric Subtraction	82.5M	36.50	2.00
Parametric Subtraction + re-ordering	82.5M	36.55	1.72

Table 4: Statistics of parameters, training and testing speeds (sentences per second). **Note that** if we don't use re-scoring strategy, the newly added parameters will never be used in testing phase. i.e., it uses the same amount of parameters as the original NMT.

Observations:

- Little increase of parameters
- Re-scoring strategy only lowers the testing speed slightly

Outline

Conclusion

In this paper, we propose to explicitly control the transition of the decoder hidden states.

- We introduce two variants to model the transitions of the decoder hidden states
- We empirically show the effectiveness of our approach in diverse language pairs

Conclusion

In this paper, we propose to explicitly control the transition of the decoder hidden states.

- We introduce two variants to model the transitions of the decoder hidden states
- We empirically show the effectiveness of our approach in diverse language pairs

Discussions

Conclusion

In this paper, we propose to explicitly control the transition of the decoder hidden states.

- We introduce two variants to model the transitions of the decoder hidden states
- We empirically show the effectiveness of our approach in diverse language pairs

Discussions

Probing methods could help us to explore what's inside the deep representations

- The better we understand the model, the more inspirations we would find.

Conclusion

In this paper, we propose to explicitly control the transition of the decoder hidden states.

- We introduce two variants to model the transitions of the decoder hidden states
- We empirically show the effectiveness of our approach in diverse language pairs

Discussions

Probing methods could help us to explore what's inside the deep representations

- The better we understand the model, the more inspirations we would find.

Simply applying re-scoring in testing helps a lot

- A better score function for the search-based decoding would be useful, which deserves further investigations.

Conclusion

Thanks!

Alexis Conneau, German Kruszewski, Guillaume Lample, Loïc Barrault, and Marco Baroni. 2018. What you can cram into a single vector: Probing sentence embeddings for linguistic properties.

Rongxiang Weng, Shujian Huang, Zaixiang Zheng, Xin-Yu Dai, and Jiajun Chen. 2017. Neural machine translation with word predictions. In *EMNLP 2017*.

Zaixiang Zheng, Hao Zhou, Shujian Huang, Lili Mou, Xinyu Dai, Jiajun Chen, and Zhaopeng Tu. 2018. Modeling past and future for neural machine translation. *TACL*, 6:145–157.