Understanding and Improving Encoder Layer Fusion in Sequence-to-Sequence Learning

Xuebo Liu[1], Longyue Wang[2], Derek F. Wong[1], Liang Ding[3], Lidia S. Chao[1], Zhaopeng Tu[2]

[1]NLP[2]CT Lab, Department of Computer and Information Science, University of Macau

[2]Tencent AI Lab [3]The University of Sydney

Introduction

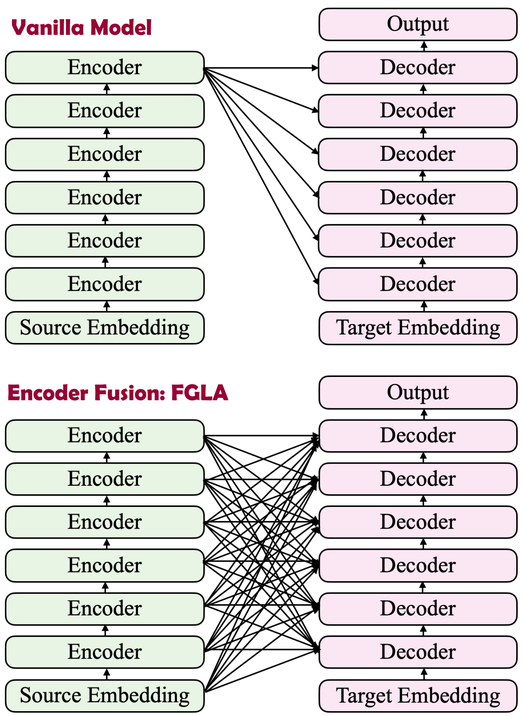
Problems

- Recent studies reveal that fusing the intermediate encoder layers (EncoderFusion) is beneficial for sequence-to-sequence (Seq2Seq) models, such as layer aggregation [1], layer-wise coordination [2] and layer attention [3]. Despite its effectiveness, not much is known about how fusing encoder layer representations work.
- Other studies show that attending to lower encoder layers (excluding the encoder embedding layer) does not improve model performance [4], which is conflicted with existing conclusions. It is still unclear why and when fusing encoder layers should work in Seq2Seq models.

Contributions

- Introduce a fine-grained layer attention method to qualitatively and quantitatively evaluate the contribution of individual encoder layers.
- Demonstrate that the encoder embedding layer is essential for fusing encoder layers, which consolidates conflicted findings reported by previous studies.
- Propose a simple yet effective SurfaceFusion approach to directly exploit the encoder embedding layer for the decoder, which produces more expressive bilingual embeddings.

Vanilla Model vs. EncoderFusion



Understanding Encoder Layer Fusion (EncoderFusion)

Source Representation Bottleneck

Seq2Seq decoder only takes the abstract representations at uppermost layer as input, which ignores other usefully surface representations at other layers. We hypothesize that its limited representation capacity may not sufficiently model those surface features from lower encoder layers, especially the embedding layer. We call such an issue as source representation bottleneck.

Fine-grained Layer Attention (FGLA)

We propose FGLA to investigate the source representation bottleneck. Specifically, FGLA replaces the layer-agnostic source representation X^N with the layer-aware representation S^m for each decoder layer Y^m:

$$\mathbf{S}^m = \sum_{n=0}^{N} \hat{\mathbf{w}}^{m,n} \odot \mathbf{X}^n, \quad \hat{\mathbf{w}}^{m,n} = \left[\hat{w}^{m,n,1}, \ldots, \hat{w}^{m,n,D}\right], \quad \hat{w}^{m,n,d} = \frac{\exp(w^{m,n,d})}{\sum_{n'=0}^{N} \exp(w^{m,n',d})}$$

where $\odot$ denotes an element-wise multiplication, and $w^{m,n,d}$ denotes an element in the learnable attention weight $\boldsymbol{W} \in \boldsymbol{R}^{M\times(N+1)\times D}$, where D is the dimensionality of the source representation.

Experimental Setup

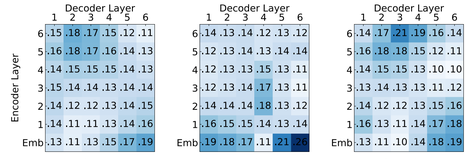
Machine translation: WMT16 Romanian-English; WMT14 English-{German, French}

Text summarization: CNN/Daily Mail

Grammatical error correction: CONLL14

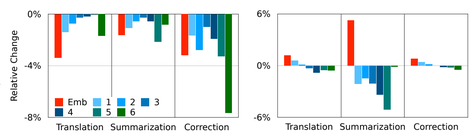
Visualization of layer attention

In all tasks, higher decoder layers especially the uppermost ones pay more attention to the encoder embedding layer, which indicates that the surface representations potentially bring some additional useful features to the model performance.



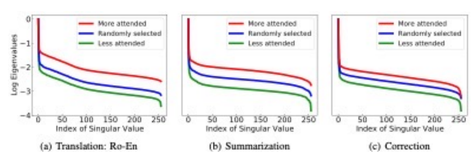
Contribution of individual encoder layer

- Masking the encoder embedding layer seriously harms the model performance in all tasks.
- Masking the encoder embedding layer consistently increases the length of generated output.



Expressivity of attended dimensions in the encoder embedding layer

Comparing the expressivity of the sub-embedding matrices by the singular value decomposition, in which higher normalized singular values denote that the embedding is more uniformly distributed, thus are more expressive. FGLA indeed learns useful surface features from the encoder embedding layer.



Improving EncoderFusion: SurfaceFusion

SurfaceFusion: Directly Exploit Source Embedding for Decoder

We rewrite $\boldsymbol{P}(y_j)$ as a fused probability with the second condition term x:

$$\hat{y} = \arg\max \prod_{j=1}^{J} \log P(y_j) = \arg\max \prod_{j=1}^{J} \log P(y_j|y_{<j}, \mathbf{x})$$

$$\log P(y_j) = \Phi\left(P(y_j|y_{<j}, \mathbf{x}),\ P(y_j|\mathbf{x})\right)$$

$$P(y_j|\mathbf{x}) = \frac{\exp(\mathbb{1}_{y_j}(\tilde{\mathbf{r}}(y_j, \mathbf{x}))/\tau)}{\sum_{w\in\mathcal{V}_y} \exp(\mathbb{1}_w(\tilde{\mathbf{r}}(y_j, \mathbf{x})/\tau)}$$

where $\Phi(\cdot)$ is a fusion method. $\boldsymbol{r}(y_j, \boldsymbol{x})$ denotes the surface feature calculated by an extra attention module and $\tilde{\boldsymbol{r}}(y_j, \boldsymbol{x})$ denotes its logit version transformed by the pre-softmax weight. We use τ to control the sharpness of the probability distribution. As τ approaches to 0, the distribution tends to be a one-hot distribution representing the token of the maximum probability. The distribution becomes uniform at a higher τ.

Choices of Fusion Function

Hard Fusion: $\Phi_{\text{hard}} = \lambda \log P(y_j|y_{<j}, \mathbf{x}) + (1-\lambda)\log P(y_j|\mathbf{x})$

Soft Fusion: $\Phi_{\text{soft}} = \log(\text{softmax}(E(y_j|y_{<j}, \mathbf{x}) + \log P(y_j|\mathbf{x}))$

where λ is an interpolation weight, and $\mathrm{E}(y_j|y_{<j}, x)$ is the pre-softmax logit of the probability $\mathrm{P}(y_j|y_{<j}, x)$. Compared to hard fusion, soft fusion removes the need for manually setting the hyperparameter λ.

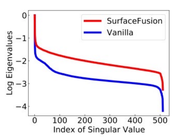
Closer Relationship between Word Embeddings

SurfaceFusion shortens the path distance between source and target embeddings, which can help to learn better bilingual embeddings with direct interactions. This can help the source and target embeddings to learn better representations, and has been validated to be beneficial for Seq2Seq models [5].

	All	Non-Shared
Vanilla	0.602	0.338
SurfaceFusion	**0.650**	**0.417**

More Expressive Word Embeddings

The singular values of SurfaceFusion become more uniformly distributed, which demonstrates that the fused surface features enhance the representation learning of embeddings.This provides a better starting point for the model to effectively extract surface and abstract features, which leads to an improvement of model performance.



Main Results

	Translation			Summarization			Correction		
	Ro-En	**En-De**	**En-Fr**	**RG-1**	**RG-2**	**RG-L**	**Prec.**	**Recall**	**$F_{0.5}$**
Existing	34.0	29.3	43.2	40.1	17.6	36.8	65.5	33.1	54.8
Vanilla	33.8	28.9	43.4	40.4	17.7	37.2	64.7	33.2	54.3
FGLA	34.5	29.1	43.5	40.8	18.0	37.5	**67.7**	31.9	55.3
Hard fusion	**35.1**	**29.5**	**43.9**	40.9	18.2	37.7	67.0	34.4	56.3
Soft fusion	34.0	29.0	43.6	**41.0**	**18.3**	**37.9**	66.8	**35.0**	**56.6**

Acknowledgement

This work was supported in part by the Science and Technology Development Fund, Macau SAR (Grant No. 0101/2019/A2), and the Multi-year Research Grant from the University of Macau (Grant No. MYRG2020-00054-FST). We thank the anonymous reviewers for their insightful comments.

References

[1] Zi-Yi Dou, Zhaopeng Tu, Xing Wang, Shuming Shi, and Tong Zhang. Exploiting deep representations for neural machine translation. In EMNLP, 2018.

[2] Tianyu He, Xu Tan, Yingce Xia, Di He, Tao Qin, Zhibo Chen, and Tie-Yan Liu. Layer-wise coordination between encoder and decoder for neural machine translation. In NIPS, 2018.

[3] Ankur Bapna, Mia Xu Chen, Orhan Firat, Yuan Cao, and Yonghui Wu. Training deeper neural machine translation models with transparent attention. In EMNLP, 2018.

[4] Tobias Domhan. How much attention do you need? a granular analysis of neural machine translation architectures. In ACL, 2018.

[5] Xuebo Liu, Derek F. Wong, Yang Liu, Lidia S. Chao, Tong Xiao, and Jingbo Zhu. Shared-private bilingual word embeddings for neural machine translation. In ACL, 2019.

* The Ninth International Conference on Learning Representations (ICLR), on May 3rd to 7th, 2021.