A. Settings for reproduction

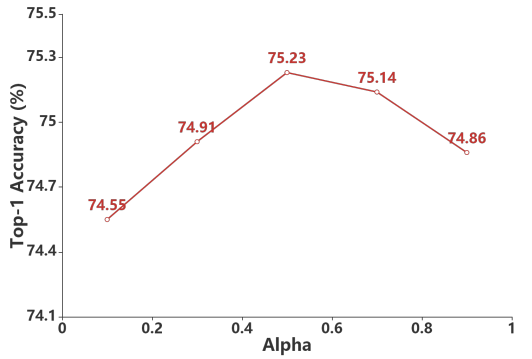
A.1. Image classification

We follow a common strategy described in Szegedy et al. (2016) for data augmentation. For ResNet (He et al., 2016), we adopt the implementation in tensorflow CNN benchmark[1]. For AA-ResNet, we modify the ResNet by augmenting 3x3 convolution with self-attention. The implementation is from the official repository of AA-ResNet[2], while we simply add the residual convolution module for EA-AA-ResNet. Concretely, we apply attention augmentation to each residual block in the last three stages – when the shapes of activation maps become 28x28, 14x14 and 7x7. We adopt the same setting as AA-ResNet, e.g. $k = 2$ and $v = 0.2$. We refer to (Bello et al., 2019) for more details. We set $\alpha = 0.5$ and $\beta = 1.0$ by default, except for EA-AA-ResNet-152 and EA-AA-ResNet-101 where we set $\alpha = 0.7$. A hyper-parameter analysis for α and β in EA-AA-ResNet-34 is shown in Figure 1. We can see that the best result is achieved at $\alpha = 0.5$ and $\beta = 1.0$, which significantly outperforms the vanilla transformer (equivalent to $\alpha = 0$ and $\beta = 0$).

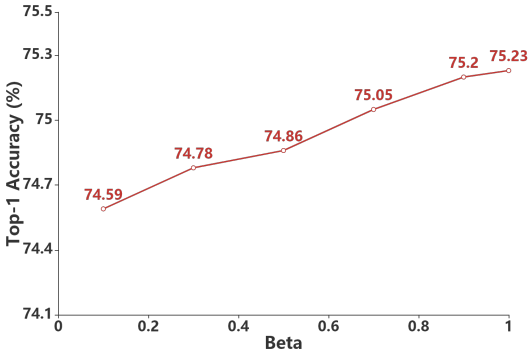
A.2. Natural language understanding

As introduced in (Devlin et al., 2019), BERT-Base and EA-BERT-Base have 12 layers and 12 attention heads with hidden dimension 768. BERT-large and EA-BERT-large have 24 layers and 16 heads, while hidden dimension for each intermediate layer is set as 1024. The hidden dimension of the final fully-connected layer before softmax is set to be 2000. We download the official checkpoints of BERT-Base[3] and BERT-Large[4], and initialize the additional parameters for EA-BERT-Base and EA-BERT-Large randomly.

We also conduct a set of experiments with RoBERTa-Large (Liu et al., 2019) and T5-Base (Raffel et al., 2019). We apply the idea of evolving attention to the network of these models. RoBERTa-Large has 24 layers with 16 attention heads. The total hidden size of all heads is 1024, and the hidden dimension of the final fully-connected layer is 4096. We use NLP library implemented by the huggingface team (Wolf et al., 2019) to implement the base version of T5, which has 220 million parameter. We download the official pre-trained checkpoints for RoBERTa-Large[5] and T5-Base[6] for fine-tuning.



(a) Analysis of α



(b) Analysis of β

Figure 1. Hyper-parameter analysis for EA-AA-ResNet-34

We adopt the Adam optimizer (Kingma & Ba, 2014) with epsilon 1e-8. The dropout rate is set as 0.1 empirically. We use grid search to optimize the values of hyper-parameters on the validation set. We search the learning rate in {1e-4, 1e-5, 2e-5}, batch size in {8, 16}, training epochs in {2, 3, 5}, α in {0.1, 0.2, 0.4} and β in {0.1, 0.2, 0.4}. The specific hyper-parameters for each task are listed in Table 1.

A.3. Machine Translation

We train the machine translation tasks using Adam optimizer (Kingma & Ba, 2014) with $\beta_1 = 0.9$, $\beta_2 = 0.98$ and an inverse square root learning rate scheduling with linear warmup. The learning rate is 1e-3 and the warmup step is set to be 4000. Also, we use label smoothing $\epsilon = 0.1$. we apply early stopping to the training procedure with a

[1] https://github.com/tensorflow/benchmarks/tree/cnn_tf_v1.14_compatible

[2] https://github.com/leaderj1001/Attention-Augmented-Conv2d

[3] https://storage.googleapis.com/bert_models/2018_10_18/uncased_L-12_H-768_A-12.zip

[4] https://storage.googleapis.com/bert_models/2018_10_18/uncased_L-24_H-1024_A-16.zip

[5] https://dl.fbaipublicfiles.com/fairseq/models/roberta.large.tar.gz

[6] https://console.cloud.google.com/storage/browser/t5-data/pretrained_models/base/

Model	Task	Training Epochs	Batch Size	Learning Rate	Adam Epsilon	Dropout Rate	α	β
	CoLA	3	8	2e-5	1e-8	0.1	0.2	0.1
	SST-2	5	8	1e-5	1e-8	0.1	0.1	0.1
	MRPC	2	8	2e-5	1e-8	0.1	0.1	0.2
	STS-B	3	8	2e-5	1e-8	0.1	0.2	0.2
BERT-Base	QQP	3	16	2e-5	1e-8	0.1	0.1	0.2
	MNLI	3	16	2e-5	1e-8	0.1	0.1	0.2
	QNLI	3	16	2e-5	1e-8	0.1	0.1	0.1
	RTE	2	8	2e-5	1e-8	0.1	0.2	0.1
	CoLA	3	8	2e-5	1e-8	0.1	0.1	0.2
	SST-2	5	8	1e-5	1e-8	0.1	0.1	0.2
	MRPC	2	8	1e-5	1e-8	0.1	0.1	0.1
	STS-B	3	8	2e-5	1e-8	0.1	0.1	0.2
BERT-Large	QQP	3	16	1e-5	1e-8	0.1	0.1	0.1
	MNLI	3	16	2e-5	1e-8	0.1	0.2	0.2
	QNLI	3	16	2e-5	1e-8	0.1	0.2	0.2
	RTE	3	16	1e-4	1e-8	0.1	0.1	0.2
	CoLA	3	8	2e-5	1e-8	0.1	0.2	0.1
	SST-2	5	8	1e-5	1e-8	0.1	0.1	0.2
	MRPC	3	16	1e-5	1e-8	0.1	0.1	0.2
	STS-B	3	16	2e-5	1e-8	0.1	0.4	0.2
RoBERTa-Large	QQP	3	16	2e-5	1e-8	0.1	0.1	0.1
	MNLI	5	16	1e-5	1e-8	0.1	0.4	0.1
	QNLI	5	16	1e-5	1e-8	0.1	0.4	0.2
	RTE	2	8	1e-4	1e-8	0.1	0.2	0.4
	CoLA	3	8	2e-5	1e-8	0.1	0.2	0.2
	SST-2	3	8	2e-5	1e-8	0.1	0.1	0.2
	MRPC	3	8	1e-4	1e-8	0.1	0.1	0.1
	STS-B	3	16	2e-5	1e-8	0.1	0.1	0.1
T5-Base	QQP	3	16	2e-5	1e-8	0.1	0.1	0.2
	MNLI	5	16	1e-5	1e-8	0.1	0.1	0.1
	QNLI	5	16	1e-5	1e-8	0.1	0.2	0.1
	RTE	3	16	1e-5	1e-8	0.1	0.4	0.2

Table 1. Detailed hyper-parameter settings for GLUE benchmark.

patience of 5 epochs. In the evaluation phase, we average the final 10 checkpoints and conduct beam search with size 5. We adopt absolute positional encoding according to the original implementation (Vaswani et al., 2017). Other hyper-parameters are optimized by grid search on the validation set and reported in Table 2.

B. Analysis

B.1. Quality of Image Attention

We select the 16th, 17th and 18th attention layers in the AA-ResNet-34 and EA-AA-ResNet34 networks for analysis. The attention maps from these layers have a shape of $14 \times 14 \times 8$, where 14 is the image length after pooling and 8 is the number of heads. Then, we send the attention maps directly as inputs to another CNN model for classification, and the original labels are used for training and evaluation. The goal is to quantify the effectiveness of attention maps learned by different models. If an attention map retains major structures of the original object, the classification accuracy should be higher. We adopt a 12-layer DenseNet (Huang et al., 2017) for attention map classification, while the shape is pooled to 7×7 and 4×4 after the 4th and 7th layers respectively. The hidden dimension is set as 256 initially and doubled after each pooling operation. The models are trained by 30 epochs with cosine learning rate decay started by 0.05 and ended by 0.0001.

B.2. Interpretability

In the context of modern neural models, attention mechanisms learn to assign soft weights to token representations, thus one can extract highly weighted tokens as rationales. In other words, if the self-attention mechanism generates better attention scores, it will achieve better performance on ERASER, a benchmark designed to evaluate the interpretability of natural language understanding models. We

Model	Task	Number of GPU	Accumulative Steps	Learning Rate	Dropout Rate	α	β
EA-Transformer-Lite	De-En	1	1	1e-3	0.2	0.1	0.1
	En-De	8	16	1e-3	0.3	0.1	0.1
	En-Fr	8	16	1e-3	0.1	0.1	0.1
EA-Transformer-Base	De-En	1	1	1e-3	0.2	0.5	0.1
	En-De	8	16	1e-3	0.3	0.5	0.1
	En-Fr	8	16	1e-3	0.1	0.5	0.1

Table 2. Detailed hyper-parameter settings for machine translation.

	Perf.↑	AUPRC↑	Comp.↑	Suff.↓
Movie Reviews				
BERT+LSTM - Attention	0.970	0.417	0.129	0.097
BERT+LSTM - EA-Attention	0.970	**0.435**	0.142	**0.084**
BERT+LSTM - Lime	0.970	0.280	0.187	0.093
EA-BERT+LSTM -Lime	**0.975**	0.313	**0.194**	0.089
FEVER				
BERT+LSTM - Attention	0.870	0.235	0.037	0.122
BERT+LSTM - EA-attention	0.870	0.238	0.078	0.097
BERT+LSTM - Lime	0.870	0.291	0.212	**0.014**
EA-BERT+LSTM -Lime	**0.886**	**0.307**	**0.236**	**0.014**
MultiRC				
BERT+LSTM - Attention	0.655	0.244	0.036	0.052
BERT+LSTM - EA-Attention	0.655	**0.251**	0.054	0.041
BERT+LSTM - Lime	0.655	0.208	0.213	-0.079
EA-BERT+LSTM -Lime	**0.674**	0.221	**0.241**	**-0.089**
CoS-E				
BERT+LSTM - Attention	0.487	0.606	0.080	0.217
BERT+LSTM - EA-Attention	0.487	**0.610**	0.113	0.189
BERT+LSTM - Lime	0.487	0.544	0.223	0.143
EA-BERT+LSTM -Lime	**0.491**	0.552	**0.231**	**0.140**
e-SNLI				
BERT+LSTM - Attention	0.960	0.395	0.105	0.583
BERT+LSTM - EA-Attention	0.960	0.399	0.177	0.396
BERT+LSTM - Lime	0.960	0.513	0.437	0.389
EA-BERT+LSTM -Lime	**0.969**	**0.534**	**0.445**	**0.368**

Table 3. Comparison of different text representation models and rationale generation methods on ERASER benchmark. "Perf." is accuracy (CoS-E) or F1 (others), AUPRC means Area Under the Precision Recall Curve; "Comp." and "Suff." denote comprehensiveness and sufficiency metrics respectively..

list the experimental results on the ERASER benchmark in Table 3. It should be noted that higher comprehensiveness scores and lower sufficiency scores are desired. First, we consider a text representation model that passes tokens through BERT and a bidirectional LSTM. Based on the same text representation, we use different methods for rationale generation. Here, the models are the same, so the downstream performance is equivalent. According to the experimental results, the rationales generated using evolving attention are more accurate than the ones generated by vanilla attention. Next, we replace vanilla BERT with EA-BERT as the text representation model and utilize a state-of-the-art rationale generation method, Lime (Ribeiro et al., 2016). Experimental results show that EA-BERT improves the performances of downstream tasks and generates better rationales simultaneously.

C. Case Study

In order to get insight into the evolving attention mechanism, we visualize exemplar attention maps for both text and image inputs and find some interesting evidences.

C.1. Image attention

In Figure 2-5, we compare the attention maps of AA-ResNet-34 and EA-AA-ResNet-34 for ImageNet classification. Compared to AA-ResNet, our proposed convolution-based evolving attention mechanism captures better global information and at the same time emphasizes on the important local information. Specifically, the residual connections and convolutional inductive bias assist the self-attention mechanism to depict a more clear outline. As shown by the visualized examples, AA-ResNet fails to compute a explainable attention map for some layers. In contrast, with the help of residual convolutions, EA-AA-ResNet successfully identifies the objects in images in an evolving process.

C.2. Text Attention

We choose BERT-Base and EA-BERT-Base models for comparison on the CoLA dataset, a task of judging the grammatical correctness of a sentence. We select the sentence "*Mary tried John to go abroad.*" for a case study. Obviously, this sentence is grammatically wrong, and a model should capture the error part "*tried John to*" in order to give the correct answer.

In Figure 6, we visualize related attention maps for three layers (#2, #11 and #12) in BERT-Base and EA-BERT-Base models. The second layer is the first layer that utilizes residual attention, and #11 and #12 are the last two layers. For each layer, we first show the attention maps from vanilla BERT and EA-BERT in the first and second columns respectively, then the convolution-based attention and self-attention maps are visualized in the third and fourth column. It should be noted that the second column is the linear fusion result of the third column and the fourth column.

Consider layer #2, both BERT (Figure 6(a)) and EA-BERT (Figure 6(b)) pay major attentions on the verb phrase "*go abroad*". As shown in Figure 6(b), EA-BERT puts additional stress on the relation between word "*tried*" and the stop sign. This is reasonable because the stop sign is responsible of capturing sentence-level semantics and "*tried*" is a key word leading to the grammatical error. As shown in Figure 6(c), the attention on this part actually comes from the convolution-based module, which is sometimes complementary to the self-attention map.

In order to ensure that the information obtained by the convolution is beneficial, we visualize the last attention layer (#12) which is the closest to the output (see Figure 6(i-l)). In Figure 6(i), we can observe that BERT-Base still focuses on verbs and stop signs in the very last layer of transformer. The attention to the wrong phrase "*tried John*" is still relatively weak, leading to a mis-classification result. In contrast, the attention scores between "*tried*" and "*John*" become obvious in EA-BERT (Figure 6(j)), largely owning to the convolutional attention map illustrated in Figure 6(k).

We also visualize the attention maps of the #11 layer, which serves as input to the #12 layer. To analysis the evolution of attention maps, we compare the differences between Figure 6(f) and Figure 6(k), as the latter is the evolved attention map taking the former as input. We find that the convolutional module helps to reason about the important word relations based on the previous attention maps. Specifically, it weakens the attention scores of the correct parts and raises higher scores for the wrong parts. As illustrated in Figure 6(k), the attention scores are significant in the upper left corner of the matrix where the error occurs. In this way, the error is fully captured in the final representation layer, helping EA-BERT to generate a correct answer.

References

Bello, I., Zoph, B., Vaswani, A., Shlens, J., and Le, Q. V. Attention augmented convolutional networks. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pp. 3286–3295, 2019.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. Bert: Pre-training of deep bidirectional transformers for language understanding. In *North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, 2019.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778, 2016.

Huang, G., Liu, Z., Van Der Maaten, L., and Weinberger, K. Q. Densely connected convolutional networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4700–4708, 2017.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., and Stoyanov, V. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692, 2019. URL http://arxiv.org/abs/1907.11692.

Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., and Liu, P. J. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv preprint arXiv:1910.10683*, 2019.

Ribeiro, M. T., Singh, S., and Guestrin, C. "why should i trust you?": Explaining the predictions of any classifier. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Demonstrations*, pp. 97–101, 2016.

Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., and Wojna, Z. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*, pp. 2818–2826, 2016.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. Attention is all you need. In *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 5998–6008, 2017.

Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., Rault, T., Louf, R., Funtowicz, M., Davison, J., Shleifer, S., von Platen, P., Ma, C., Jernite, Y., Plu, J., Xu, C., Scao, T. L., Gugger, S., Drame, M., Lhoest, Q., and Rush, A. M. Huggingface's transformers: State-of-the-art natural language processing. *ArXiv*, abs/1910.03771, 2019.

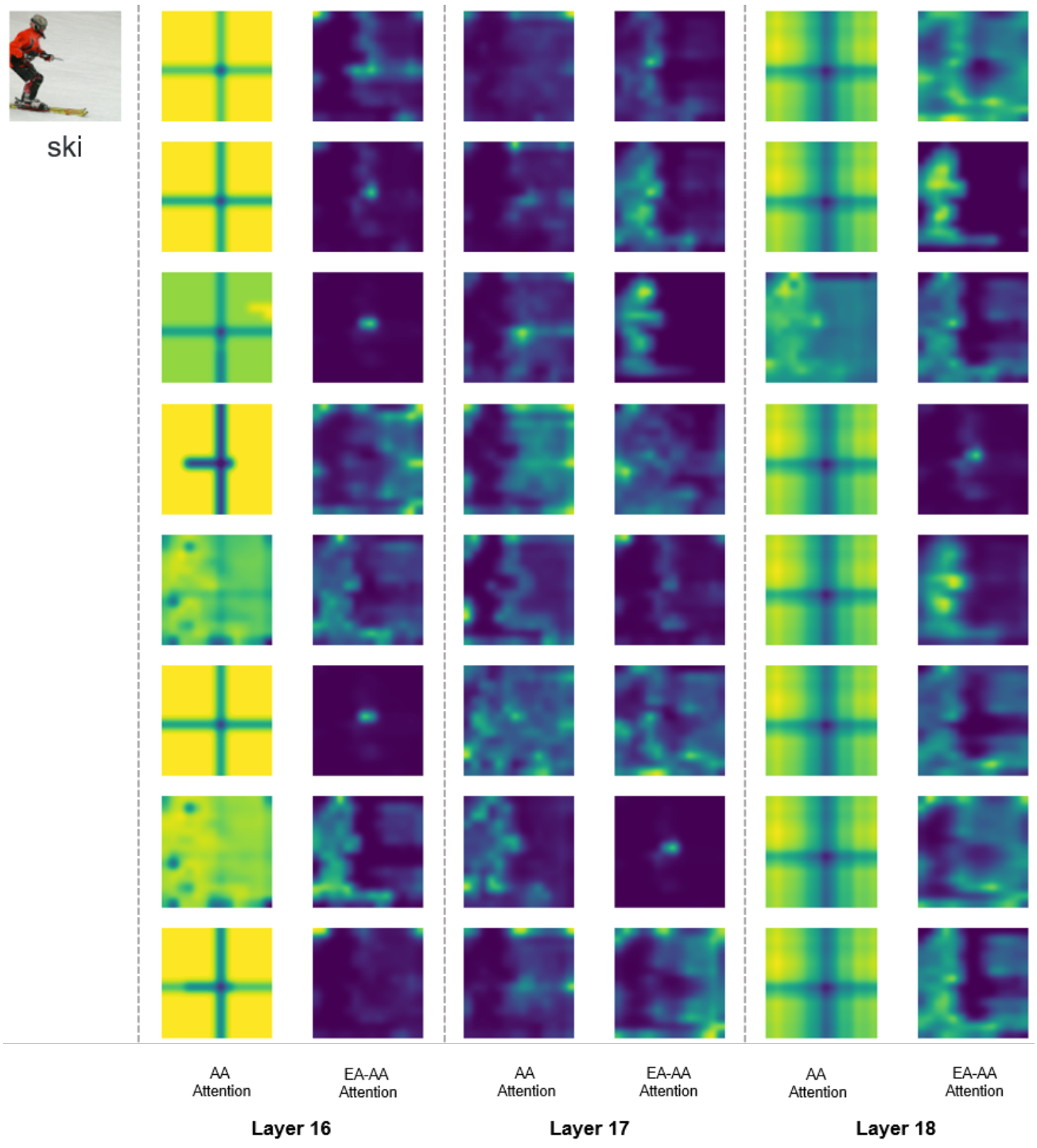


Figure 2. Attention map visualization for an image classification example

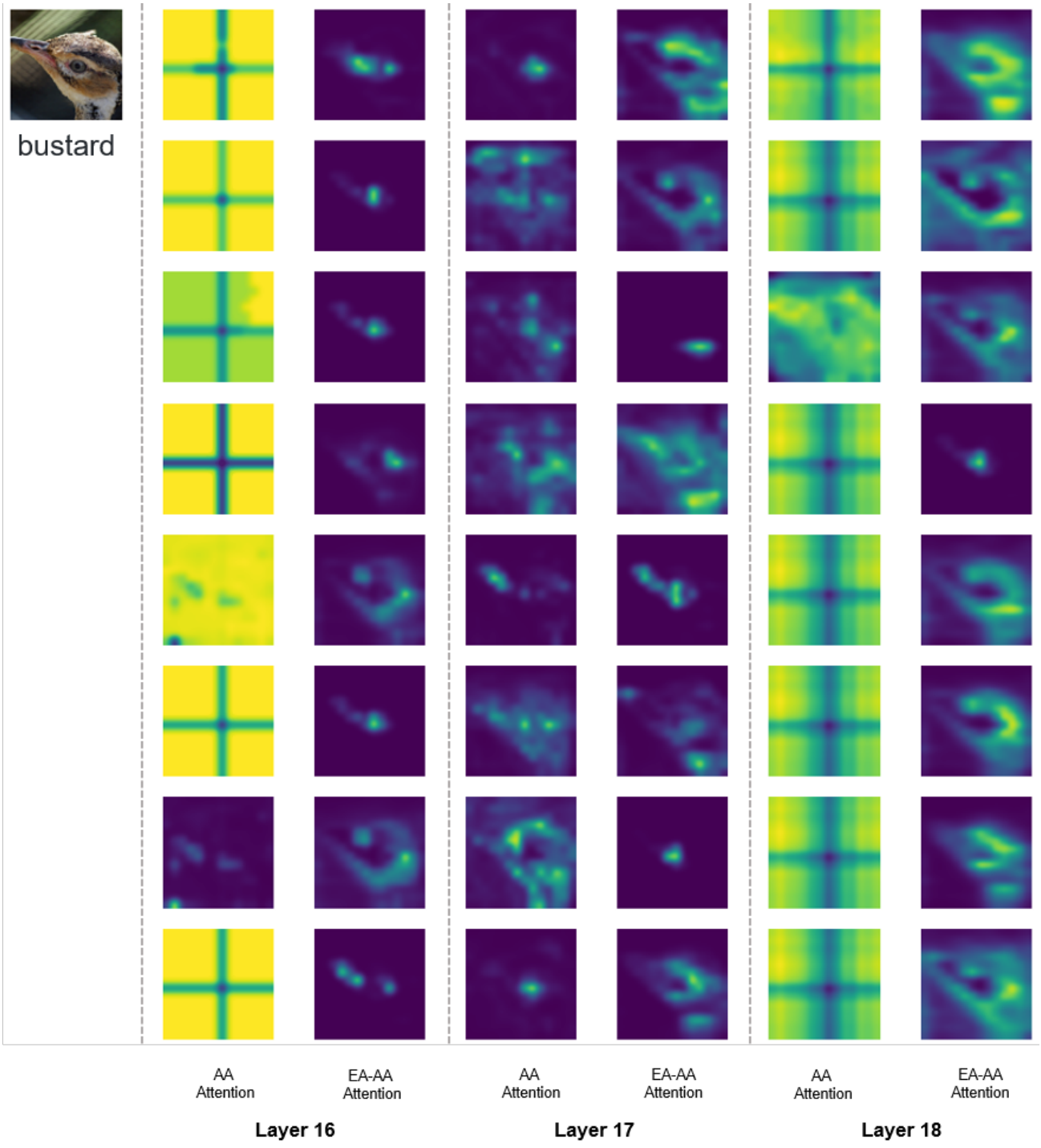


Figure 3. Attention map visualization for an image classification example

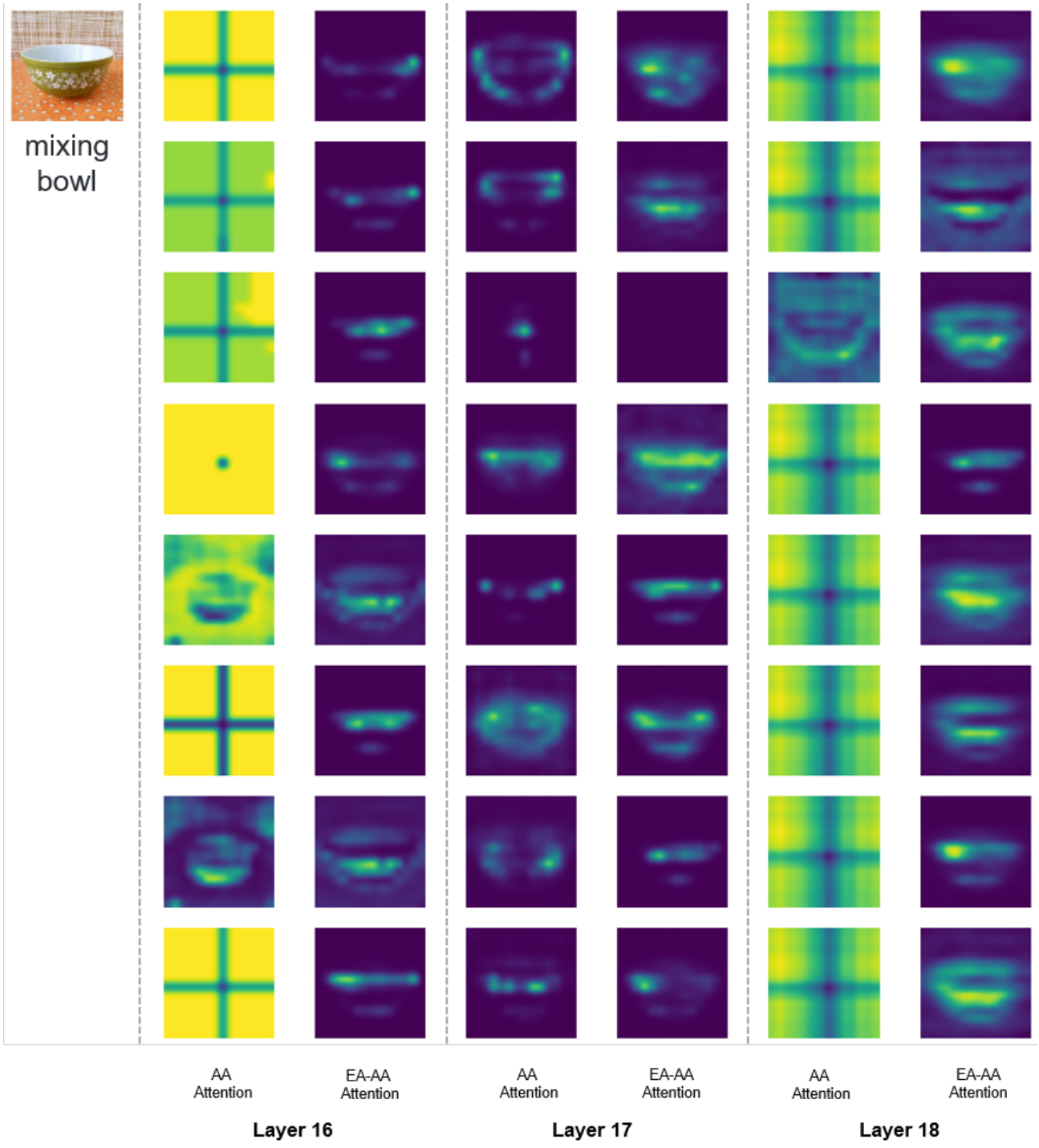
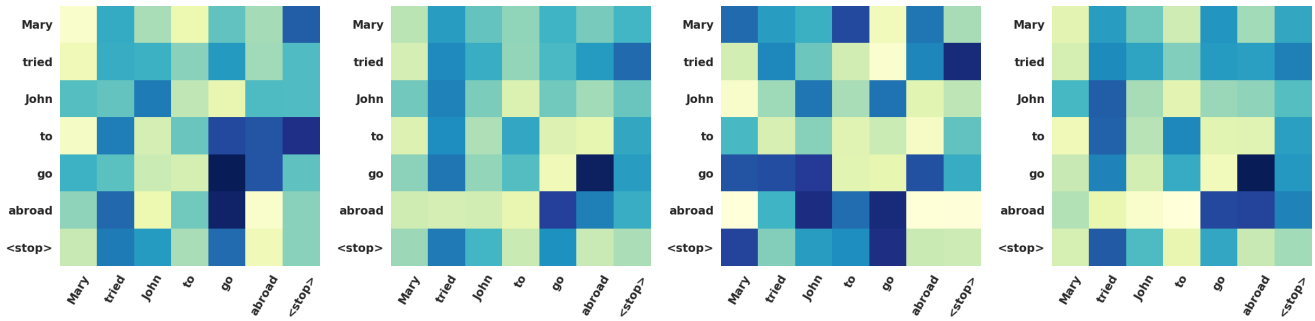
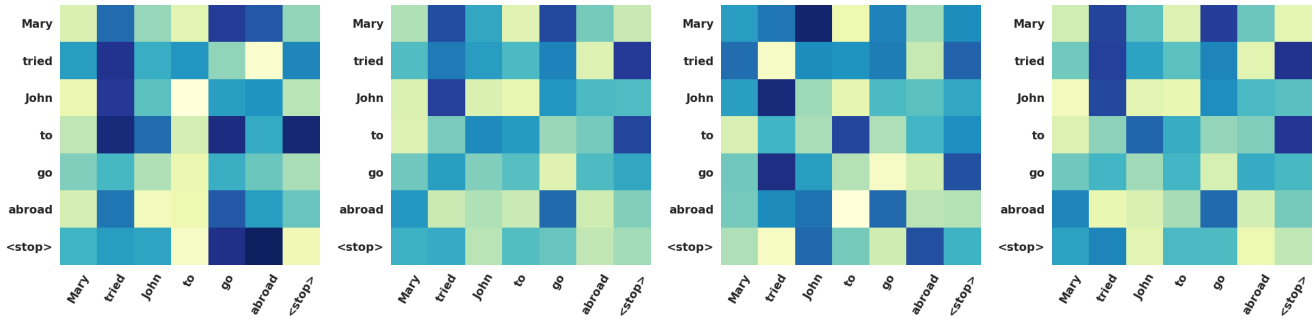


Figure 4. Attention map visualization for an image classification example

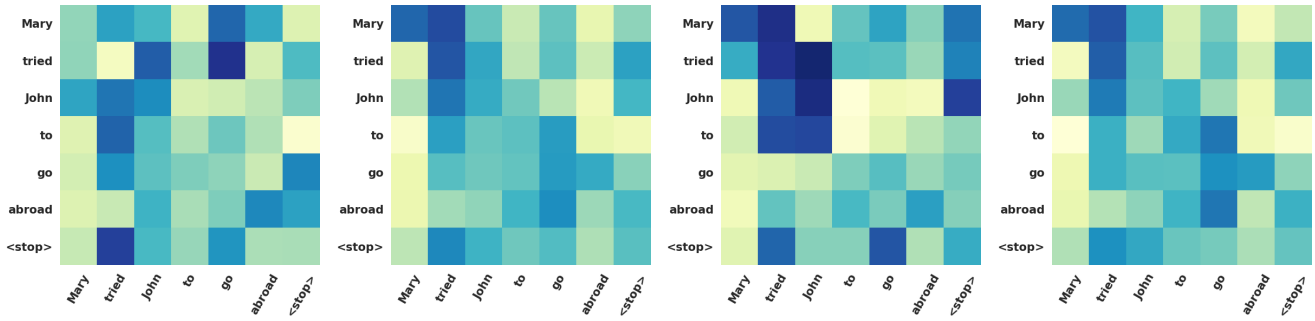
Figure 5. Attention map visualization for an image classification example



(a) BERT #2 (b) EA-BERT #2 (c) Convolution #2 (d) Self-Attention #2



(e) BERT #11 (f) EA-BERT #11 (g) Convolution #11 (h) Self-Attention #11



(i) BERT #12 (j) EA-BERT #12 (k) Convolution #12 (l) Self-Attention #12

Figure 6. Attention maps of layer #2, #11 and #12 for "*Mary tried John to go abroad.*"