AtMan: Understanding Transformer Predictions Through Memory Efficient Attention Manipulation

Maciej Chrabąszcz

XAI for transformer-based models

Through their increasing size, transformers are particularly challenging for explainability methods, especially for architecture-agnostic ones.

The attention matrix dimensions match that of the input sequence dimension, which makes the attention mechanism in particular suited for deriving input explanations. Consequently, most explainability adoptions to transformers focus on the attention mechanism.

However, while being efficient, it often emphasizes irrelevant tokens, in particular, due to its classagnostic nature.

Influence Functions as Explainability Estimators

Given a sequence w = [w1, . . . , wN], an AR language model assigns a probability to that sequence p(w) by applying factorization. The loss optimization during training can then be formalized by solving:

$$\max_{\theta} \log p_\theta(\mathbf{w}) = \sum_t \log p_\theta(w_t | w_{<t})$$

$$= \sum_t \log \text{softmax}(h_\theta(w_{<t}) W_\theta^T)_{target^t} \quad (1)$$

$$=: -\sum_t L^{target}(\mathbf{w}_{<t}, \theta)$$

$$=: L^{target}(\mathbf{w}, \theta) \; . \quad (2)$$

Influence Functions as Explainability Estimators

Perturbation methods study the influence of the model's predictions by adding small noise to the input and measuring the prediction change

$$\mathcal{I}^{target}(z_\epsilon, z) = \frac{dL^{target}(z, \theta_\epsilon)}{d\epsilon}\Big|_{\epsilon=0}$$
$$\approx L^{target}(z, \theta_{-z_\epsilon}) - L^{target}(z, \theta)\ . \qquad (3)$$

Single Token Attention Manipulation

The core idea of ATMAN is the shift of the perturbation space from the raw input space to the embedded token space. This allows authors to reduce the dimensionality of possible perturbations down to a single scaling factor per token.

$$\widetilde{\mathbf{H}}_{u,*,*} = \mathbf{H}_{u,*,*} \circ (\mathbf{1} - \mathbf{f}^i), \qquad (4)$$

Then they define target explanation as the vector of the influen

$$\mathcal{E}(\mathbf{w}, target) := (\mathcal{I}^{target}(\mathbf{w}_1, \mathbf{w}), \ldots, \mathcal{I}^{target}(\mathbf{w}_n, \mathbf{w})) ,$$

$$\mathcal{I}^{target}(\mathbf{w}_i, \mathbf{w}) := L^{target}(\mathbf{w}, \theta_{-i}) - L^{target}(\mathbf{w}, \theta) .$$

Correlated Token Attention Manipulation

Suppressing single tokens works well when the entire entropy responsible to produce the target token occurs only once. However, for inputs with redundant information, this approach would often fail.

Authors applied cosine similarity ***s*** in the embedding space, e.g., right after the embedding layer, which is known to be a good correlation distance estimator.

$$\mathbf{f}^{i}_{k,*} = \begin{cases} s_{i,k}, & \text{if } k \neq i, \kappa \leq s_{i,k} \leq 1, \\ \kappa, & \text{if } k \neq i, s_{i,k} \leq \kappa, \\ 0, & \text{otherwise.} \end{cases}$$

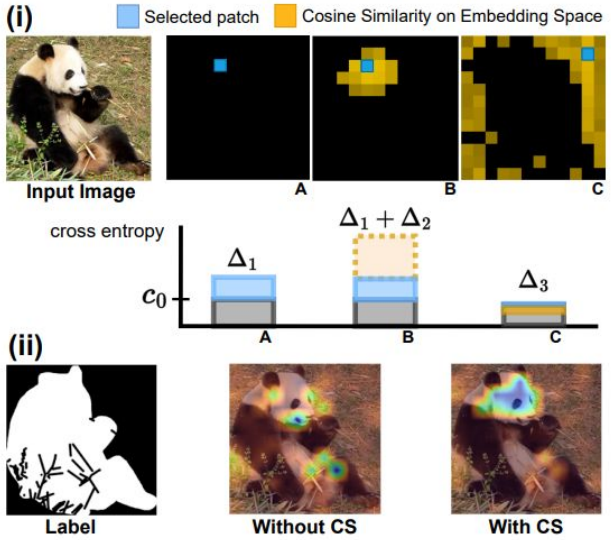
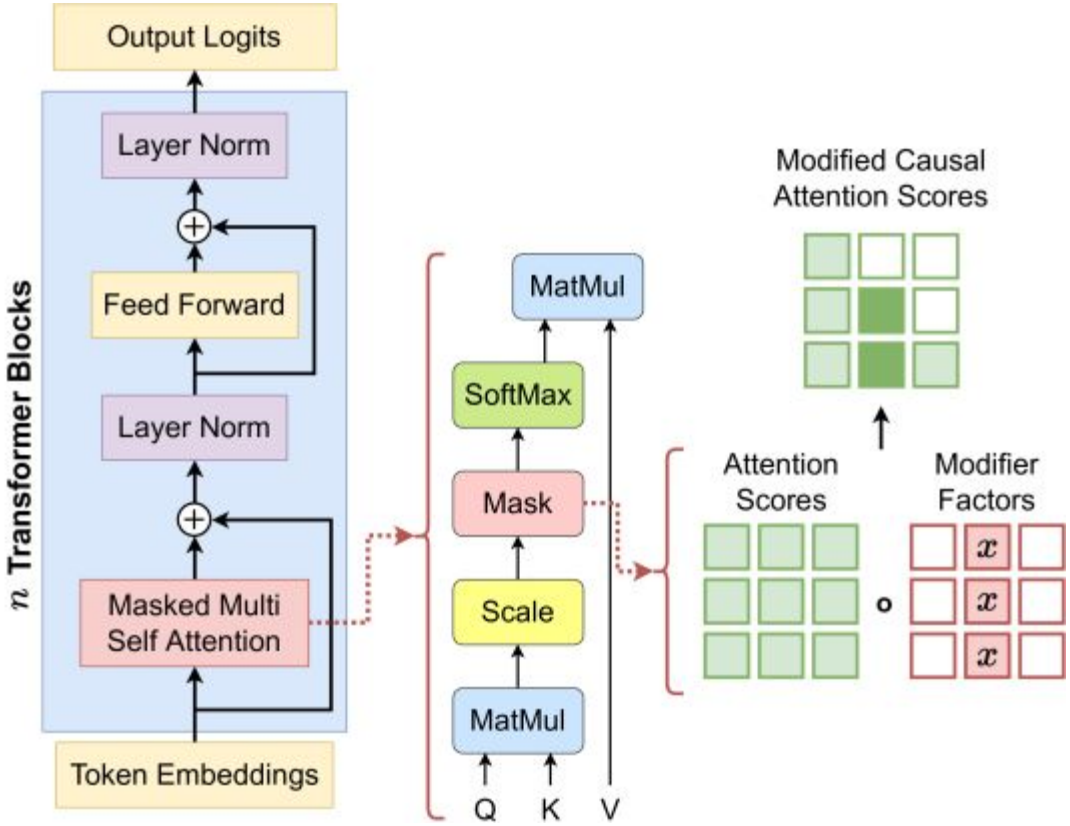


Figure 5: **Correlated token suppression of ATMAN enhances explainability in the image domain.** i) Shows an input image along with three perturbation examples (A, B, C). In A we only suppress a single image token (blue), in B the same token with its relative cosine neighborhood (yellow). In C a non-related token with its neighborhood. Below are depicted the changes in cross-entropy loss. c_0 is the original score for the target token "panda". Δ denotes the loss change. ii) Shows the label, the resulting explanation without Cosine Similarity (CS) and with CS. (Best viewed in color.)

AtMan


Output Logits
Layer Norm
Feed Forward
Layer Norm
Masked Multi Self Attention
n Transformer Blocks
Token Embeddings
MatMul
SoftMax
Mask
Scale
MatMul
Q
K
V
Modified Causal Attention Scores
Attention Scores
Modifier Factors
x
x
x

Empirical Evaluation

	IxG	IG	Chefer	ATMAN
mAP	51.7	49.5	∘72.7	•73.7
mAP_{IQ}	61.4	49.5	∘77.5	•81.8
mAR	91.8	87.1	•96.6	∘93.4
mAR_{IQ}	•100	98.6	•100	•100

Table 1: **ATMAN outperforms XAI methods on the QA dataset SQuAD.** Shown are (interquartile) mean average precision and mean average recall (the higher, the better). Best and second best values are highlighted with • and ∘.

	IxG	IG	GradCAM	Chefer	ATMAN	$ATMAN_{30B}$
mAP	38.0	46.1	56.7	49.9	•65.5	∘61.2
mAP_{IQ}	34.1	45.2	60.4	50.2	•70.2	∘65.1
mAR	0.2	0.3	0.1	11.1	•13.7	∘12.2
mAR_{IQ}	0.1	0.1	0.1	10.1	•19.7	∘14.5

Table 2: **ATMAN outperforms XAI methods on the VQA benchmark of OpenImages.** Shown are (interquartile) mean average precision and mean average recall (the higher, the better). Best and second best values are highlighted with • and ∘. XAI methods are evaluated on a 6B model, except the last column, in which case only ATMAN succeeds in generating explanations.

Empirical Evaluation

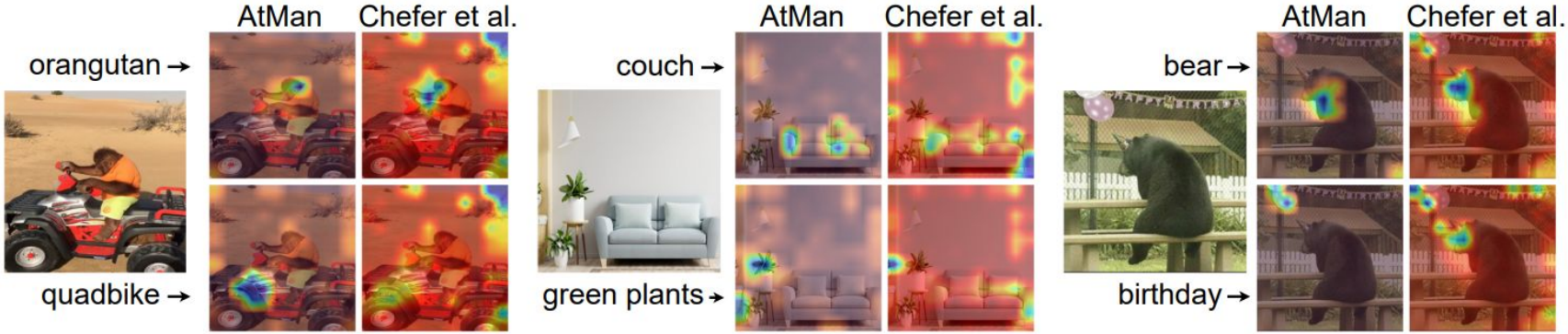


Figure 7: **ATMAN produces less noisy and more focused explanations when prompted with multi-class weak segmentation** compared to Chefer. The three shown figures are prompted to explain the target classes above and below separately. It can be observed that both methods produce reasonable, and even similar output. Though more sensitivity and more noise is observed on the method of Chefer. In particular on the last example, for the *target* "birthday", Chefer highlights more details like the decoration. However the same is also derived to some extent when just prompting "bear". (Best viewed in color.)

AtMan at scale

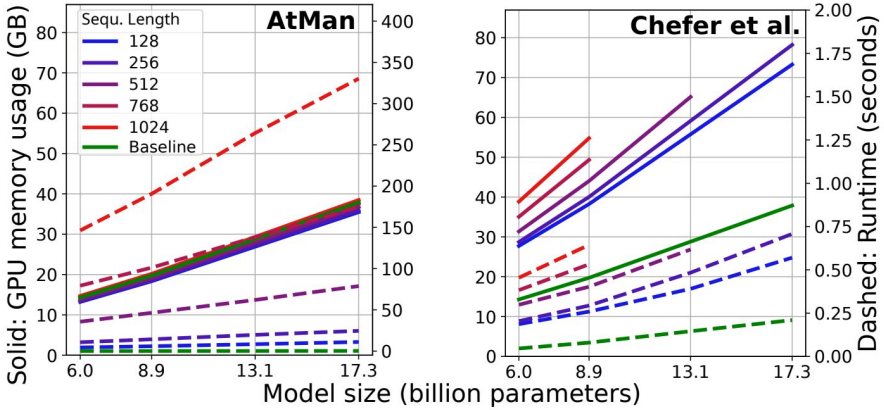


Figure 8: **AtMan scales efficiently.** Performance comparison of the explainability methods AtMan and Chefer *et al.* over various model sizes (x-axis) executed on a single 80GB memory GPU. Current gradient-based approaches do not scale; only AtMan can be utilized on large-scale models. Solid lines refer to the GPU memory consumption in GB (left y-axis). Dashed lines refer to the runtime in seconds (right y-axis). Colors indicate experiments on varying input sequence lengths. As baseline (green) a plain forward pass with a sequence length of 1024 is measured. (Best viewed in color.)

Any Questions?