

Supplementary Materials: Trust Prophet or Not? Taking a Further Verification Step toward Accurate Scene Text Recognition

Anonymous Authors

1 THE DETAILED INFERENCE FOR SPE

We denote patch embedding as $x_i \in \mathbb{R}^c$ for the patch P_i at scale i . Here, we ignore the superscript label j of x_i for a general patch embedding representation. Each value $x_i^{(k)}$, $k \in \{1, \dots, c\}$ in the vector x_i is computed by:

$$x_i^{(k)} = \langle x_i, w^{(k)} \rangle = F(P_i)^T F(w^{(k)}), \quad (1)$$

where $w^{(k)} \in R^{w_i \times h_i \times C}$ are the patch embedding weights, $\langle \cdot, \cdot \rangle$ denotes the dot product, and F is the flatten operation turning the multi-dimensional array into a vector.

If the patch region is resized to a different scale P_{i^*} , its flattened result can be represented by a linear transformation:

$$F(P_{i^*}) = B_i^{i^*} F(P_i), \quad (2)$$

where $B_i^{i^*} \in \mathbb{R}^{w_{i^*} \times h_{i^*} \times C}$ is the parameter of the linear mapping from scale i to i^* . We ignore the channel C because we resize each channel independently.

We expect to find a “resized” patch embedding weights \hat{w} , so the patch representations before and after resizing remain the same, i.e., $\langle x_i, w \rangle = \langle x_{i^*}, \hat{w} \rangle$. The problem can be formulated as the optimization function:

$$\begin{aligned} \hat{w} &= \arg \min_{\hat{w}} \mathbb{E}_{x_i \sim D} [\langle x_i, w \rangle - \langle Bx_i, \hat{w} \rangle]^2 \\ &= \arg \min_{\hat{w}} \mathbb{E}_{x_i \sim D} [(x_i^T (w - B^T \hat{w}))^2] \\ &= \arg \min_{\hat{w}} \mathbb{E}_{x_i \sim D} [(w - B^T \hat{w})^T x_i (x_i^T (w - B^T \hat{w}))] \\ &= \arg \min_{\hat{w}} (w - B^T \hat{w})^T \mathbb{E}_{x_i \sim D} [x_i x_i^T] (w - B^T \hat{w}), \end{aligned} \quad (3)$$

$B = B_i^{i^*}$ and D is the distribution over the patches.

We use the symbol Σ to represent $\mathbb{E}_{x_i \sim D} [x_i x_i^T]$, which is the covariance matrix of D . The minimization of Eq. (3) could be further transformed to:

$$\hat{w} = \arg \min_{\hat{w}} (\sqrt{\Sigma} (w - B^T \hat{w}))^T (\sqrt{\Sigma} (w - B^T \hat{w})). \quad (4)$$

Then, we have $\hat{w} = (\sqrt{\Sigma} B^T)^+ \sqrt{\Sigma} w$. $(M)^+$ is the pseudo-inverse of matrix M . When the patch distribution $D \sim \mathcal{N}(0, I)$, \hat{w} is equal to $(B^T)^+ w$. We can observe that the matrix to transform the patch embedding weights w for a certain scale i corresponds to the inverse of the bilinear resize operation B . Therefore, we calculate the normalized patch embedding at any scale i^* as:

$$x_{i^*}^{(k)} = F(P_{i^*}) (B_i^{i^*})^+ F(w^{(k)}). \quad (5)$$

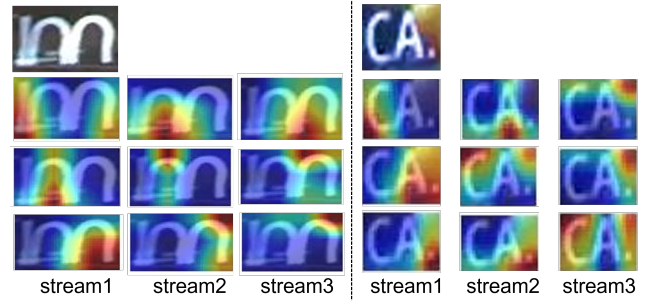


Figure 1: Attention of incorrect recognized text in different streams of PD₃. The image of “m” is recognized as “inn”, and “CA” is recognized as “cat”.

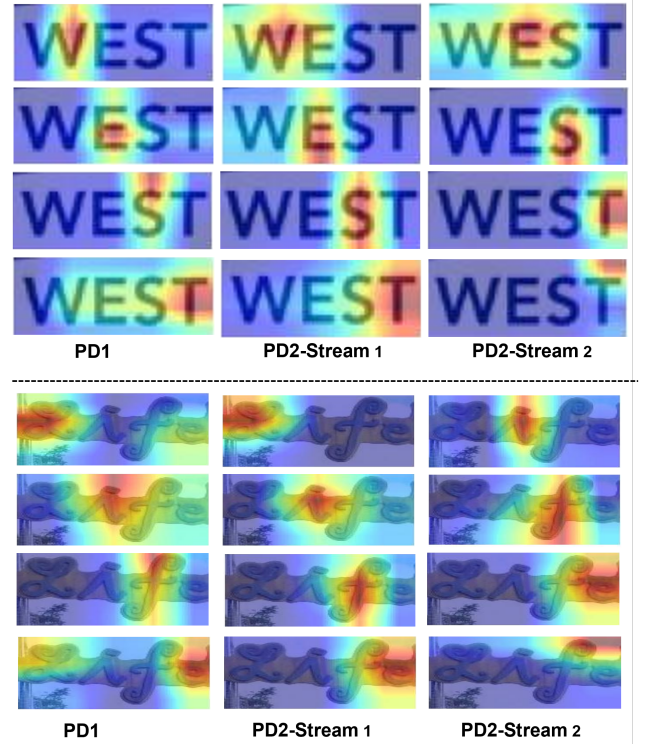


Figure 2: Attention of horizontal text in different prophet decoders.

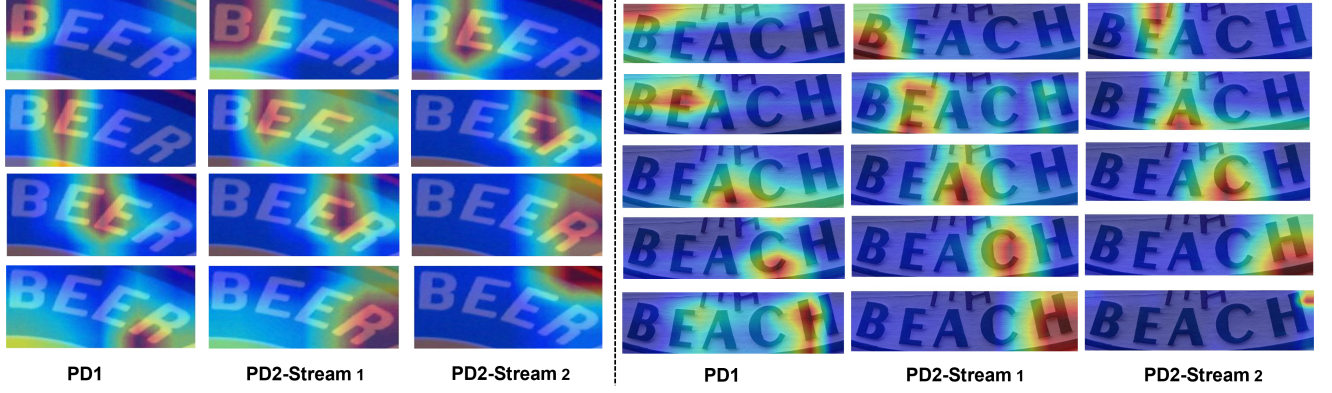


Figure 3: Attention of rotated text in different prophet decoders.

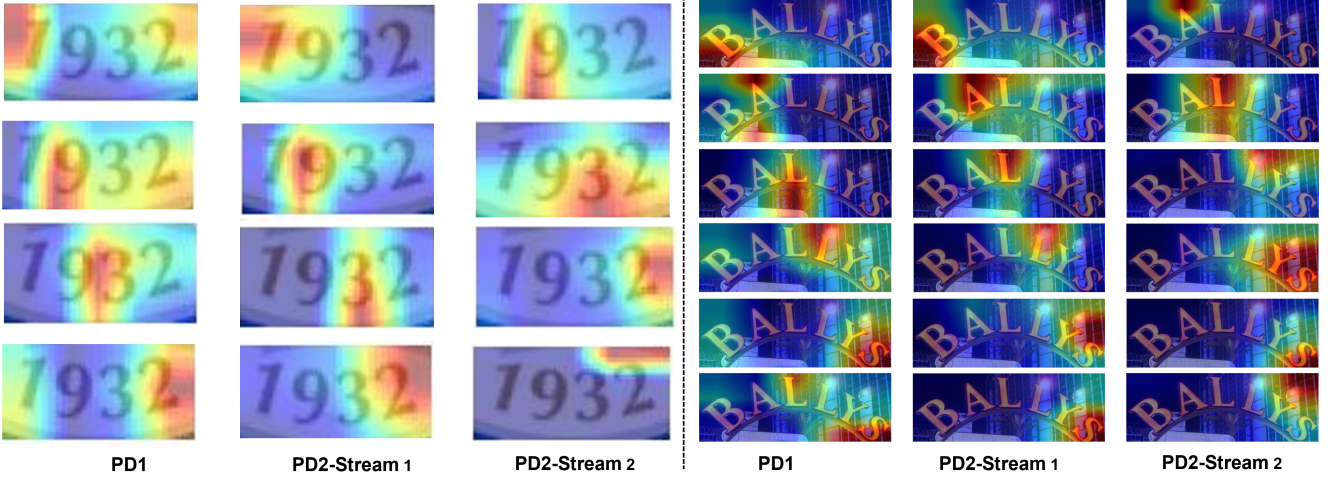


Figure 4: Attention of curve text in different prophet decoders.

2 MORE VISUALIZED ATTENTION OF PROPHET DECODER

We visualize more characters' attention by using different prediction streams in the decoder. Firstly, we visualize the character attention in each stream of PD₃ for incorrectly recognized text in Fig. 1. The t_{th} row displays the attention at time step $t-1$. As analyzed in Section 4.3.2, the gap between models with PD₂ and PD₃ is not large. It is mainly caused by the wrong attention on the short text with the increasing streams in PD.

Then, we fix the single-scale encoder and connect it with single stream prophet decoder (i.e., PD₁) and two-stream prophet decoder (i.e., PD₂) to get the character's attention in the two decoders at each time step respectively. The detailed Visualization results are shown in Fig. 2. The character attention in the first stream of PD₂ is more accurate than that of PD₁ which only has one prediction stream. The second stream of PD₂ could also pay high attention correctly to the next character during prediction. It could promote

the concentration of character attention for the first stream. This can be observed for both horizontal text, rotated text, and curved text in Fig. 2, 3 and 4 respectively.

3 CORRECTED RESULTS BY MVM

We present the models with MVM and without MVM in Table 5 of Section 4.3.3. It boosts the average performance on benchmark datasets by 1.58%. We further evaluate the models w and w/o MVM module on WordArt and Union14M-Benchmark respectively. MVM could improve the averaged accuracy by 2.2% for the WordArt dataset and 1.3% for the Union14M-Benchmark. The inference time of MVM is about 3ms, which illustrates its significant performance with meager computational cost. More corrected results by MVM on low contrast text (1_{st} row), artistic text (2_{nd} row), rotated text (3_{rd} row), curved text (4_{th} row), text with complex background (5_{th} row) and multi-linguistic text (last row) are displayed in Fig. 5.

						
PD2-stream1:	register	esplanada	estd	aurt	fonmula	snapon
PD2-stream2:	eglster</s>	splarada</s>	sto</s>	ubt</s>	onmula</s>	napan</s>
MVM:	register	esplanade	estp	hurt	formula	snapon
GT:	register	esplanade	estp	hurt	formula	snapon
						
PD2-stream1:	boba	tron	ltars	dobbe	uultimate	bienna
PD2-stream2:	obo</s>	roh</s>	tans</s>	abb</s>	ltimate</s>	rewna</s>
MVM:	boba	tron	stars	dabbe	ultimate	brenna
GT:	boba	tron	stars	dabbe	ultimate	brenna
						
PD2-stream1:	crawlord	soov	sumosal	g20	college	mod
PD2-stream2:	nawfosd</s>	ool</s>	umos8</s>	2o</s>	ollec</s>	ov</s>
MVM:	crawford	soon	sumosal	g20	college	mod
GT:	crawford	soon	sumosal	g20	college	mod
						
PD2-stream1:	chelsea	ballack	kolind	motors	macic	cuisine
PD2-stream2:	helsed</s>	allach</s>	oling</s>	otoas</s>	agic</s>	ulsine</s>
MVM:	chelsea	ballack	kolind	motors	macic	cuisine
GT:	chelsea	ballack	kolind	motors	macic	cuisine
						
PD2-stream1:	sept2	dreamer	jweet	eragon	safanis	cocacola
PD2-stream2:	eptz</s>	heamer</s>	weet</s>	rogod</s>	alaris</s>	ocalola</s>
MVM:	sept2	dreamer	sweet	eragon	safaris	cocacola
GT:	sept2	dreamer	sweet	eragon	safaris	cocacola
						
PD2-stream1:	marilyn	retlaites	electricol	awaysioe	masterfile	giahk
PD2-stream2:	orilgn</s>	etraltes</s>	lcctrical</s>	wayside</s>	astenfile</s>	iafk</s>
MVM:	marilgn	retraites	electrical	awayside	masterfile	giahk
GT:	marilgn	retraites	electrical	awayside	masterfile	giahk

Figure 5: The character recognition results of different streams in prophet decoder and MVM.