# Supplementary material for
# Joint Visual Semantic Reasoning: Multi-Stage Decoder for Text Recognition

Ayan Kumar Bhunia[1] Aneeshan Sain[1,2] Amandeep Kumar* Shuvozit Ghose*
Pinaki Nath Chowdhury[1,2] Yi-Zhe Song[1,2]
[1]SketchX, CVSSP, University of Surrey, United Kingdom.
[2]iFlyTek-Surrey Joint Research Centre on Artificial Intelligence.
{a.bhunia, a.sain, p.chowdhury, y.song}@surrey.ac.uk {shuvozit.ghose, kumar.amandeep015}@gmail.com.

## A. Optimal number of stages:

In any multi-stage network, it is quite natural for performance to saturate after a few stages. For us, it saturates after s = 2. It is well accepted in parallel literature that adding more layers to convolutional network does not necessarily improve the model's performance even if representation power is increased theoretically. Moving forward, ResNet was designed to handle very deep networks, however even for ResNet [3], adding more layers often saturates or even lowers performance in various tasks. Therefore, our empirical observation is not a surprise and is aligned with the findings of other parallel works. This performance saturation could be attributed to vanishing gradient problem which is addressed via residual/dense connection, but still persists to some extent. Also, for $s > 2$, the joint visual-semantic information might reach its optimum, where the result saturates.

## B. Use of Gumbel-Softmax and its novelty:

Designing the Gumbel-Softmax [2] is not actually our novelty. We deal with the idea that in order to exploit the semantic information for text recognition, prediction should be refined in a stage-wise manner. However, characters are discrete tokens, thus we emulate Gumbel-Softmax operation to enable gradient flow across-stages. Here, we fixed the temperature parameter to 1.0, as temperature annealing failed to improve performance. The major contribution here however, is how we design a multi-stage framework that is unrolled in a stage-wise manner and meets the objective of visual-semantic reasoning tailor made for text recognition.

## C. Latency of the recognition modules:

Any multi-stage framework would have at least had a bit higher latency. For instance, convolutional pose machine [1] being a multi-stage framework is widely accepted by the community for pose estimation. Nevertheless, we perform a study where we use a ResNet-101 as backbone feature extractor to make the number of parameters and flops of baseline model very close to ours. However, the performance still lags by $8 - 9\%$ compared to ours. Therefore, our performance gain is not tied to higher computational cost as any naive stacking of multiple layers can not attain any performance gain. We conclude that well-motivated and careful designs (*stage-wise unrolling*) lead to this performance boost.

## D. Justification of the multiple constraints:

The core motivation of multiple constraints came from multi-task learning, where multiple tasks are constrained to accelerate the learning process and reduce the training time. Similarly, the two losses $L_V$ and $L_S$ are used as auxiliary losses for driving towards better convergence that enriches individual character aligned feature, with better visual and semantic information.

*Interned with SketchX

## E. Why the proposed method has reasoning capability?

Let's take a word image with annotation 'aeroplane'. In case of single stage attentional decoder, if the model wrongly predicts 'n' instead of 'r', 'aen' would have adverse effect on the rest of the prediction, and due to single stage prediction it would have no choice to refine the prediction. Furthermore, while predicting the first few characters, it has almost negligible semantic context information. If we unroll the prediction in a stage wise manner, and a character is predicted wrongly, like 'aenoplane', it captures the semantic information from previous prediction. This aids in refinement during the later stages coupled with visual information (Eq. 3). Speaking intuitively, previous stage prediction 'aenoplane' could help refine 'n' to 'r' exploiting semantic reasoning.

## F. Background on Transformer [8] used for Visual-Semantic Reasoning:

A transformer network consists of encoder blocks, each comprising several layers of multi-head attention [8] followed by a feed forward network. The encoder is usually designed as a stack of N identical layers, each of which has two sub-layers, a multi-head self-attention (MHA) and a feed-forward network (FFN). Contrary to traditional sequence modeling methods (RNN/LSTM) which learns the temporal order of current time steps from previous steps (or future steps in bidirectional encoding), the attention mechanism in transformers allows the network to decide which time steps to focus on to improve the task at hand. A single-head self-attention (SHA) module $\texttt{attn}(Q, K, V)$ is defined as:

$$\texttt{attn}(Q, K, V) = \texttt{softmax}(\frac{QK^T}{\sqrt{d_k}})V \tag{1}$$

Accordingly other network components are defined as:

$$\begin{aligned} \text{SHA}_i(Q, K, V) &= \texttt{attn}(QW_i^Q, KW_i^K, VW_i^V) \\ \text{MHA}(Q, K, V) &= \texttt{concat}(\text{SHA}_1, ..., \text{SHA}_h)W^O \end{aligned} \tag{2}$$

$$\text{FFN}(x) = \max(0, xW_1 + b_1)W_2 + b_2 \tag{3}$$

where Q, K and V are *Query*, *Key* and *Value* respectively, $\text{SHA}_i$ is the $i$-th attention head. FFN $(\cdot)$ is composed of two linear transformations with ReLU in between where $W_{(\cdot)}^{(\cdot)}$ are learnable matrices. Around each of the sub-layers, a residual connection with layer normalization is placed. Therefore, the output of each sub-layer is $\texttt{LayerNorm}(x + \texttt{sublayer}(x))$. The decoder is also designed as a stack of N identical layers following [8] with sub-layers, – *two* MHA sub-layers followed by 1 FFN. The additional MHA sub-layer performs multi-head *self-attention* on the encoder's output. Similar to encoder stack, residual connections and layer normalizations are applied to each sub-layer as well.

## G. Handling curved texts:

Instead of using rectification network [7, 9], we use 2D attention [4] that focuses on localized character regions to predict every character. Even if the text is curved or irregular, 2D attention manages to discover the sequential character-location in a weakly supervised manner, as shown in [4]. Therefore, we attribute the better performance on curved text to our multi-scale2D attention module which can deal with characters of varying shapes within the curved text.

## H. Clarity on faster convergence:

While using dense and residual connections, we notice that the loss mostly saturates after 300K iterations with lesser fluctuations, however upon removing them, we notice it takes way more number of iterations (around 500K) to saturate the loss along with a dip in the performance. This re-verifies the necessity of dense and residual connections for faster convergence of the model.

## I. Handling deletion/insertion error:

We use stage-wise unrolling here. Every stage is unrolled afresh, taking information from prior stages. Therefore, even if some deletion/insertion errors were to happen in the earlier stages, the number of characters in the later stages would *not* need to follow the same, as the end token will be predicted separately for successive stages. In other words for example, even if there are 5 characters predicted in stage-0, the later stages can have either higher or less number of characters, thus enabling us to deal with deletion-insertion errors from prior stages. Please see the examples in Fig. 3 of the paper.

## J. Details on ResNet architecture:

We use the ResNet architecture from [5, 6], where both use the same back-bone CNN. The code is taken from implementation of [5]. We only modify the stride of pooling layer to obtain feature map from last three layers of spatial sizes 4x25,8x25, and 16x50.

## K. An idea of Computation time:

For comparison, we evaluate the computation time for ASTER [7] – one of the seminal works for text recognition. It takes 20.21 ms as compared to ours of 26.31 ms under the same experimental conditions. Please note that ASTER does not involve multi-stage decoding, but includes an additional text rectification network. Considering this, we argue that our multi-stage decoding adds minimal computational overhead and overall time is within a reasonable remit.

## L. Additional Experimental Inferences:

(i) We conducted a study where we fed the concatenated visual and semantic feature vectors to a single transformer. However, instead of any gain, the performance lags by 3% on IC15 compared to ours. Therefore, we argue that information passing across the time steps needs to occur within an individual modality, followed by visual-semantic fusion.
(ii) We tried combining results from multiple stages using simple mean operation at every time step of decoding. However, resulting performance decreased by 5.7% on IC15. We attribute this to the poor performance of earlier stages. Consequently, when prior stages' prediction are combined with that of final stage, it corrupts the overall result as well.

## References


[1] Mingfei Gao, Mingze Xu, Larry S Davis, Richard Socher, and Caiming Xiong. Startnet: Online detection of action start in untrimmed videos. In *ICCV*, 2019. 1

[2] Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. In *ICLR*, 2017. 1

[3] Sergey Karayev, Tobias Baumgartner, Mario Fritz, and Trevor Darrell. Timely object recognition. *NeurIPS*, 2012. 1

[4] Hui Li, Peng Wang, Chunhua Shen, and Guyu Zhang. Show, attend and read: A simple and strong baseline for irregular text recognition. In *AAAI*, 2019. 2

[5] Canjie Luo, Lianwen Jin, and Zenghui Sun. Moran: A multi-object rectified attention network for scene text recognition. *PR*, 2019. 3

[6] Baoguang Shi, Xinggang Wang, Pengyuan Lyu, and Cong Yao. Robust scene text recognition with automatic rectification. In *CVPR*, 2016. 3

[7] Baoguang Shi, Mingkun Yang, Xinggang Wang, Pengyuan Lyu, Cong Yao, and Xiang Bai. Aster: An attentional scene text recognizer with flexible rectification. *T-PAMI*, 2018. 2, 3

[8] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, 2017. 2

[9] Kai Wang, Boris Babenko, and Serge Belongie. End-to-end scene text recognition. In *ICCV*, 2011. 2