ATTENTIVE SEQUENTIAL NEURAL PROCESSES

Anonymous authors
Paper under double-blind review

ABSTRACT

Sequential Neural Processes (SNP) is a new class of models that temporally extend Neural Processes (NP) and can meta-learn a sequence of stochastic processes. This learned function however under-fits the provided contexts as is also the case in NP. Applying attention to contexts serves as a solution but doing the same in SNP is challenging because it cannot directly access and attend over the past contexts. In this paper, we propose Attentive Sequential Neural Processes (ASNP) which resolves the under-fitting in SNP by introducing a novel imaginary context, modeled as a latent variable, over which the attention can then be applied. We evaluate our model on 1D Gaussian Processes regression and 2D moving MNIST/CelebA regression. We apply ASNP to implement Attentive Temporal-GQN and further evaluate on the moving CelebA task.

1 INTRODUCTION

Neural Processes (NP) combine the strengths of neural networks and Gaussian processes (GP) such that, like Gaussian processes, it can flexibly learn a new stochastic process while at test time still providing fast $\mathcal{O}(1)$ prediction speed like neural networks. Learning from small datasets from different tasks (i.e., different stochastic processes), NP can also be seen as a probabilistic latent variable framework for meta-learning. Sequential Neural Processes (SNP) (Singh et al., 2019) are a class of sequential latent generative models that extend the power of neural processes (NP) to a sequence of stochastic processes. SNP targets problems where the sequence of stochastic processes are governed by an underlying transition dynamics and thus learning to transfer this temporal trend between stochastic processes are useful. Combining the meta-learning aspect of NP with temporal transfer, SNP can be seen as a meta-transfer learning model (i.e., transfer learning of meta learning) or (temporal) stochastic process of stochastic processes.

A well-known problem of NP is underfitting: summarizing all context data into a single latent vector by a permutation-invariant encoding, it is quite easy to lose the details of the individual datapoint in the context. To resolve this problem, Kim et al. (2019a) proposed Attentive Neural Processes (ANP). In ANP, a query-dependent representation is obtained by applying an attention mechanism to the context data points with a query data as the key. Therefore, when making a prediction on a query, the model utilizes not only the global summary of the NP latent but also is informed by the query-sensitive representation in which the details of individual data points are provided through attention, mitigating the underfitting problem.

Providing a single global latent for all the past and current context, SNP is also subject to the underfitting problem, perhaps more severely. One may consider applying attention to SNP in a similar way as ANP does. However, as a sequential model which, unlike ANP, does not store any of the past context but compresses them into a single encoding through an RNN, it is quite unclear how to realize an attention mechanism in SNP. Although we can still apply attention limitedly to the context of the current time step, in SNP we assume this context is a very small amount and even empty. Therefore, at first glance, augmenting SNP with an attention mechanism may seem somewhat an ill-posed problem.

In this paper, we propose to resolve this problem by introducing imaginary context that augments the context set, as latent variables, and then apply attention on the union of the real context and inferred latent context. We call the proposed model as Attentive Sequential Neural Processes (ASNP). ASNP consists of four modules: (i) context imagination, (ii) global context-encoding without attention, (iii) local attentive context-encoding, and (iv) prediction. ASNP uses attention at two-levels. It first

obtain the imaginary observations using attention and then make the prediction using attention over the imaginary context given a query. In experiments, we demonstrate the effectiveness of applying this attention mechanism into SNP for several scenarios of a variety of tasks: 1D Gaussian Process Regression, 2D moving MNIST/CelebA Regression and 2D moving CelebA rendering.

The contributions of the paper are as follows. We first introduce a new model, Attentive Sequential Neural Processes, that resolves the problem of augmenting attention mechanism on SNP. Then, we demonstrate that the proposed model can resolve the underfitting problem in SNP in various scenarios and tasks.

2 BACKGROUND

Neural Processes (NP) (Garnelo et al., 2018b) learns the process of learning a stochastic process, mapping an input $x \in \mathbb{R}^{d_x}$ to a random variable $y \in \mathbb{R}^{d_y}$, using a set of context samples $C = (X_C, Y_C) = \{x_i, y_i\}_{i \in \mathcal{I}(C)}$. Here, $\mathcal{I}(C)$ is the set of indices for the elements in set C. Specifically, to learn a stochastic process from a context, NP uses the conditional prior $P(z|C)$ and then it can use the representation $z \sim P(z|C)$ of the inferred stochastic process to make prediction y on query x. This is modeled by $p(y|x, z)$. The full generative process of NP can be written as:

$$P(Y|X, C) = \int P(Y|X, z)P(z|C)\mathrm{d}z \tag{1}$$

where $P(Y|X, z) = \prod_{i \in \mathcal{I}(D)} P(y_i|x_i, z)$ and D is the *target* observations $D = (X, Y) = \{x_i, y_i\}_{i \in \mathcal{I}(D)}$. The training data for NP is obtained by iterating sampling a stochastic process and then sampling (C, D) from the sampled stochastic process. Thus, considering a stochastic process as a task, NP can be seen a probabilistic meta-learning framework.

The **Generative Query Networks** (GQN) (Eslami et al., 2018) are an application of NP to the problem of learning representation and rendering of 3D scenes from a set of partial 2D observations. In GQN, a query x becomes a camera viewpoint in a 3D environment and an output y corresponds to an image taken from the viewpoint x. The scene representation of the original GQN is query-dependent and thus can make inconsistent generations across queries and thus in the following we use Consistent GQN (CGQN) (Kumar et al., 2018) that resolves this problem by making the scene representation query-independent. For simplicity, in the following our use of the term GQN actually refers CGQN.

The **Attentive Neural Processes (ANP)** are proposed to resolve the underfitting problem of NP by (Kim et al., 2019b). To this end, ANP has two context-encoding paths. One path is the same as the scene representation of NP summarizing all context information independently to the target query x. The other path is attentive encoding of the context which is dependent to the target query x due to its usage as the attention query. Because the detail information of specific contexts can be preserved with attention, this results in mitigating the underfitting problem of NP.

Sequential Neural Processes (SNP) proposed by (Singh et al., 2019) introduce temporal transition of underlying stochastic processes. Thus, SNP can be seen as a stochastic process of stochastic processes or meta-transfer learning. This transition modeling is done by introducing temporal prior $P(z_t|z_{<t}, C_t)$ that depends on (i) the progress of the past stochastic processes from which we learn the general temporal trend and (ii) the context of the current time step which provides a quick meta-learning ability for the current stochastic process. Then, given the updated representation z_t, we can make predictions in the same way as in NP, i.e., $P(y_t|x_t, z_t)$. Combining these and extending to a time-horizon, we have the following generative model of SNP:

$$P(Y, Z|X, C) = \prod_{t=1}^{T} p(y_t|x_t, z_t)p(z_t|z_{<t}, C_t). \tag{2}$$

Applying SNP to the problem of GQN, Singh et al. (2019) also proposes Temporal Generative Query Networks (TGQN). As an extension of NP toward temporal modeling, SNP is also prone to the underfit the context as does NP. Therefore, it can be seen as a natural direction to apply attention to SNP to resolve the underfitting problem.

3 ATTENTIVE SEQUENTIAL NEURAL PROCESSES

How can we augment SNP with an attention mechanism? This problem might look simple at first glance, provided that there is already the ANP model available, is actually not quite straightforward. This is because, unlike the ANP which assumes all context points are stored in a memory, SNP assumes that it cannot store the past context as it is. This means that, in SNP the entities to apply attention on is severely limited only to the context of the current time-step. This is problematic because we assume that the context per step is small or even empty. In such a setting, attention is not effective in general.

To resolve this problem, we introduce the *imaginary context*, and thereby augment the entities on which attention can be applied. This imaginary (or pseudo) context observations are treated as latent variables and are generated by an RNN that encodes all of the past context, both the real and the imaginary. Thus, the role of the imaginary context is to infer "*which data points other than the real context would be helpful if we were able to refer to those for attention and prediction?*" Similarly, it can also be seen as a dynamic and conditional restoration (decoding) process of context after summarizing (encoding) the past through an RNN. After this restoration, we can apply attention on the extended context set. In the following, we now describe the proposed model ASNP with particular focus on (i) how to generate the imaginary context and (ii) how to apply attention thereafter.

3.1 GENERATIVE PROCESS

The generative process of ASNP can be decomposed into four modules: (i) context imagination, (ii) global context-encoding, (iii) local (attentive) context-encoding, and (iv) prediction. The purpose of **context-imagination** at each step t is to generate an imaginary context set $C_t^i = (X_t^i, U_t^i)$ which consists of imaginary queries $X_t^i = \{x_{t,k}^i\}_{k=1}^K$ and its corresponding observations (or its representation) $U_t^i = \{u_{t,k}^i\}_{k=1}^K$. That is, we treat these imaginary context as latent variables. The generation is conditioned on the real context C_t^r and all the past contexts $C_{<t}$ (where C_t is the combined context $C_t^r \cup C_t^i$) through an RNN encoding. That is, it uses the current real context as a query to generate an imaginary context. The generation of imaginary observation is conditioned on the imaginary query, and attention over the real context is used. We explain this attention in more detail in the next section. Then, the whole context-imagination model can be written as:

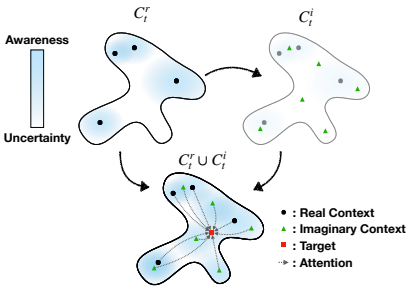

Figure 1: Illustration about imaginary context. Conditioned real context C_t^r, imaginary context is generated. Here, the imaginary queries causes lower uncertainty to locate on important points.

$$P(C_t^i|C_{<t}, C_t^r) = \prod_{k=1}^{K} P(x_{t,k}^i, u_{t,k}^i|C_{<t}, C_t^r) = \prod_{k=1}^{K} P(u_{t,k}^i|x_{t,k}^i, C_{<t}, C_t^r) P(x_{t,k}^i|C_{<t}, C_t^r). \quad (3)$$

An interesting interpretation on this modeling can be observed in comparison to the memory retrieval process in the human brain (Eichenbaum, 2017). As it is well-known in neuroscience, "*human memory is not a literal reproduction of the past, but instead relies on constructive processes that are sometimes prone to error and distortion*" (Schacter, 2012). Our model resembles this process in the sense that it (i) compresses the past observation and importantly what it has imagined in the past through an RNN (which can be considered as lossy memory consolidation), and then after observing current context, (ii) recalls from the compressed memory not a simple copy of the past experiences but a constructive recreation of representations that is directed to help prediction.

In **global context-encoding**, we obtain the global representation z_t which provides the representation of underlying stochastic process independently to queries. This module can be considered the prior in the standard SNP where we do not use attention and can be written as $P(z_t|z_{<t}, C_t)$. Note that it is a design choice whether to choose C_t or C_t^r for the conditioning context of this module. If we use C_t^r, z_t becomes global context-encoding of real contexts. Otherwise, it becomes encoding of the combined contexts. The **local context-encoding** is a query-dependent encoding using attention over the combined context. Here, we use the query x_t as the attention query, and the $(x, y) \in C_t$

pairs in the combined context as the key-value slot to attend. This is a deterministic function that gives us the attention encoding $r_{x_t} = f_{\text{attend}}(x_t, C_t)$ which we describe in more detail in the next section. Lastly, the **prediction model** $P(y_t|x_t, z_t, r_{x_t})$ can be implemented simply by providing additional vector r_{x_t} to the likelihood function SNP. Combining all, we can write the full generation process of ASNP as follows:

$$P(Y, Z, C^i|X, C^r) = \prod_{t=1}^{T} P(y_t|x_t, z_t, C_t)P(z_t|z_{<t}, C_t)P(C_t^i|C_{<t}, C_t^r). \tag{4}$$

3.2 Attention with Imagination

In ASNP, we use attention in two modules. The first attention is to obtain the imaginary output C_t^i. For this, we first generate X_t^i from an RNN that takes the previous imaginary inputs X_{t-1}^i and the context encoding $v_t^r = f_{\text{order-inv}}(C_t^r)$ as the input and updates its state to h_t^X. Then, X_t^i is sampled from h_t^X, implementing $\prod_{k=1}^{K} P(x_{t,k}^i|C_{<t}, C_t^r)$. Given this generated imaginary input X_t^i, the imaginary output-encoding $u_{t,k}^i$ is generated as follows. First, for each imagination $k \in \{1, \dots, K\}$, we maintain an imagination-tracker RNN, which is denoted by $h_{t,k}^u$ and takes $(x_{t,k}^i, u_{t-1,k}^i)$ as input. Using the union of the imaginary context-pairs and the real context-pairs, $S_t^i = \{(x_{t,k}^i, h_{t,k}^u)\} \cup \{(x_t^r, f_{y\to u}(x_t^r, y_t^r))\}$, as the key-value set, we apply attention on the set S_t^i with $x_{t,k}^i$ as the attention query, and obtain the attention encoding $a_{t,k}^i$. Then, we sample $u_{t,k}^i \sim \mathcal{N}(f_\mu^u(a_{t,k}^i), f_\sigma^u(a_{t,k}^i))$ and complete the module $P(u_{t,k}^i|x_{t,k}^i, C_{<t}, C_t^r)$.

Algorithm 1 Attention with Imagination

$h_t^X = \text{RNN}_X((X_{t-1}^i, v_t^r), h_{t-1}^X)$
$X_t^i \sim \mathcal{N}(f_\mu^X(h_t^X), f_\sigma^X(h_t^X))$
$h_{t,k}^u = \text{RNN}_u((x_{t,k}^i, u_{t-1,k}^i), h_{t-1,k}^u)$
$a_{t,k}^i = f_{\text{attend}}(x_{t,k}^i, S_t^i)$
$u_{t,k}^i \sim \mathcal{N}(f_\mu^u(a_{t,k}^i), f_\sigma^u(a_{t,k}^i))$
$r_{x_t} = f_{\text{attend}}(x_t, C_t)$

The second attention is implemented in the prediction module $P(y_t|x_t, z_t, C_t)$. Given x_t as the attention query, we use C_t, the union of the real C_t^r and imagined C_t^i context, as the key-value set (after encoding each context $x \in X_t^r$ and $y \in Y_t^r$ using $f_{y\to u}$). The attention encoding $r_{x_t} = f_{\text{attend}}(x_t, C_t)$ is then concatenated with the global encoding z_t to make an input for the prediction y_t. Algorithm 1 describes an algorithm about the imagination update and attention encoding.

3.3 Learning and Inference

Due to the intractability of the true posterior, ASNP is trained via variational approximation with the following posterior approximation:

$$P(Z, C^i|C^r, D) \approx \prod_{t=1}^{T} Q(z_t|z_{<t}, C_t^i, C^r, D)Q(C_t^i|C_{<t}^i, C^r, D) \tag{5}$$

where $Q(C_t^i|C_{<t}^i, C^r, D) = Q(U_t^i|X_t^i, C_{<t}^i, C^r, D)Q(X_t^i|C_{<t}^i, C^r, D)$ and $D = (X, Y)$. For training, the following evidence lower bound (ELBO) is maximized w.r.t. θ and ϕ:

$$\begin{aligned}\mathcal{L}_{\text{ASNP}}(\theta, \phi) = &\sum_{t=1}^{T} \mathbb{E}_{Q_\phi(z_t, C_t^i|C^r, D)} \left[\log P_\theta(y_t|x_t, z_t, C_t^i, C_t^r)\right] \\ &- \mathbb{E}_{Q_\phi(z_{<t}, C_{<t}^i)} \left[\mathbb{KL}\left(Q_\phi(z_t, C_t^i|C^r, D) \parallel P_\theta(z_t, C_t^i|z_{<t}, C_{<t}^i, C_t^r)\right)\right].\end{aligned} \tag{6}$$

For backpropagation, we use reparameterization trick (Kingma & Welling, 2013) for continuous variables. Derivation of eq. 6 is in Appendix A.

4 Related Works

Modeling stochastic process on few data set is attractive recently related with meta-learning. Conditional Neural Processes (CNP) (Garnelo et al., 2018a) models a regression function without global latent, which causes inconsistency between queries. NP (Garnelo et al., 2018b) resolves it by proposing an explicit global latent. While CNP and NP are designed for more simple task, GQN (Eslami

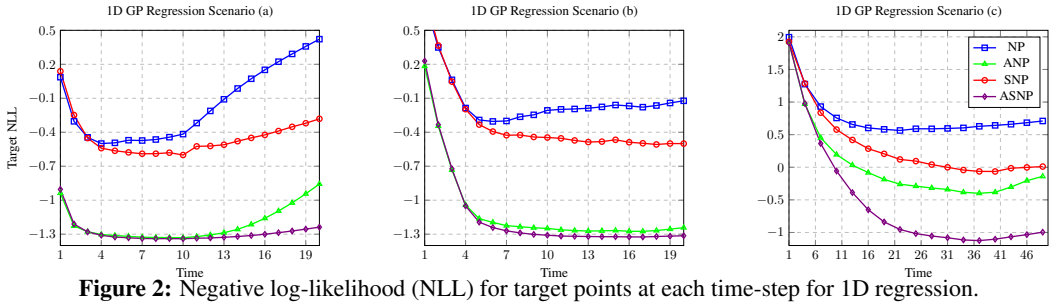

Figure 2: Negative log-likelihood (NLL) for target points at each time-step for 1D regression.

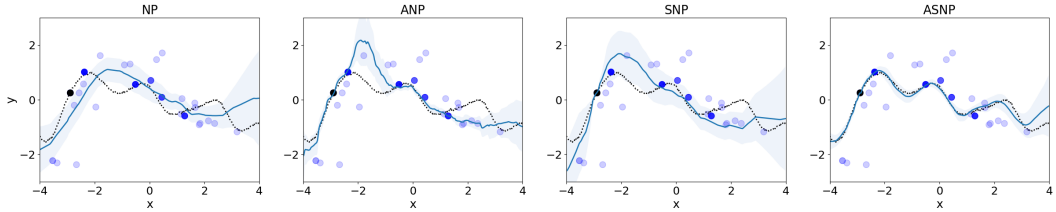

Figure 3: Sample for 1D regression task (c) at t=33. Dot line is groundtruth, and blue line is the mean of prediction. Sky-blue area means uncertainty. Black dot is context at t=33 and more dense blue dot is more recent context.

et al., 2018) develops it to render 3D scene. Like CNP, GQN is also inconsistency between queries due to absent of a global latent and CGQN (Kumar et al., 2018) resolves it with a global latent. In a line to enhance performance, ANP (Kim et al., 2019b) resolves the underfitting with attention for queries and Rosenbaum et al. (2018) applies attention on GQN model by attention given raw images as a query to render complex 3D scene (i.e. Minecraft). SNP (Singh et al., 2019) is proposed as a generalized model for sequential stochastic processes with TGQN for rendering the dynamic 3D scene. Functional Neural Processes (FNP) (Louizos et al., 2019) is proposed to learn a graph of dependencies between pre-selected points and training points for modeling distributions over functions.

In aspect that imaginary context is pseudo and few number of trainable representative points, it is related to inducing points of Set Transformer (Lee et al., 2019) and sparse GPs (Snelson & Ghahramani, 2006; Titsias, 2009). VampPrior (Tomczak & Welling, 2017) also uses few trainable pseudo-inputs to resolve overfitting, over-regularization and high computational complexity. The reference set of FNP to model a graph of small number of points is also motivated from inducing points of sparse GPs. In aspect of using trainable memory, the differential neural dictionary (Pritzel et al., 2017) stores and updates the previous trajectories as (key, value) pair to learn fast a variety of environments.

5 Experiments

We evaluate ASNP on the following selection of tasks: *a)* 1D Gaussian Process (GP) regression *b)* 2D moving MNIST (LeCun et al., 1998) and 2D moving CelebA (Liu et al., 2015). On these tasks, we evaluate and compare the proposed model against NP, ANP, SNP as the baselines. Each task is evaluated with three scenarios, which is used in (Singh et al., 2019) for 1D GP regression: (a) To evaluate the model's ability to extrapolate the future, we provide high-amount of context in the first 10 time-steps out of the 20. (b) To test the model on its ability to track the dynamics using intermittently arriving context, we provide high-amount of context in 10 randomly chosen time-steps out of the 20. (c) To test the model's ability to gather and make use of low amount of context received over long segments of time, we provide low amount of context in 45 randomly chosen time-steps out of the 50. In each setting, the remaining time-steps are used for prediction. The details about model architectures can be found in Appendix B.1.

To test the benefits of attention for rendering 2D images, we used moving CelebA because it has higher uncertainty when given partial knowledge in comparison to an alternative data set such as moving simple shapes (e.g. circle). We test on the scenario (c) as described above, but with shorter

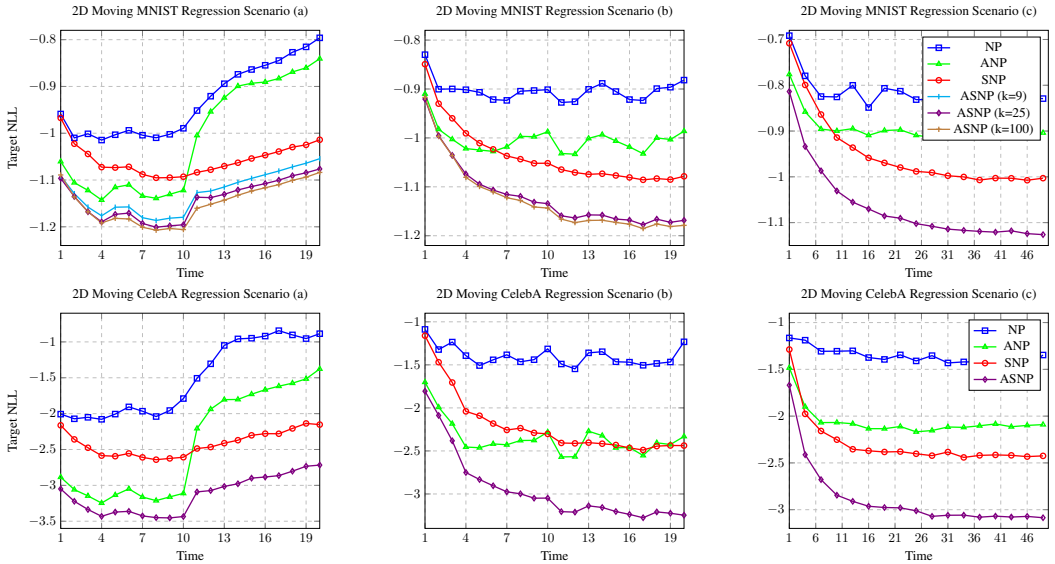

Figure 4: Target NLL at each time-step for 2D moving MNIST (**Top**) and CelebA (**Bottom**) regression.

sequence lengths and smaller context sizes. More details are explained in the section on 2d rendering.

1D regression: We first evaluated our method on sequential 1D Gaussian Process data set. This is an extension of the GP data set through the addition of linear dynamics on the kernel hyper-parameters as used in (Singh et al., 2019). In each transition, we add a small Gaussian noise to introduce stochasticity. For scenarios (a) and (b), the number of context observations n is randomly selected in $[5, 50]$ or an empty set. The number of target observations m is randomly selected in $[1, 51 - n]$. For scenario (c), n is 1 or 0, and m is chosen from $[1, 11 - n]$. The target set subsumes the context. The more details about this task is described in Appendix C.

In Fig 2, SNP outperforms NP as reported in (Singh et al., 2019). On the other hand, in scenario (a), the performance of ANP degrades steeply after the context are removed since it does not model the dynamics explicitly. Even though it appears to outperform NP and SNP, this can be credited to the attention in ANP that prevents underfitting (Kim et al., 2019b). ASNP shows better performance for all the scenarios, outperforming ANP with a greater gap in task (c). We hypothesize that when dense context is given at each time-step in (a) and (b), ANP can fit well as it does not need to rely on the dynamics much. But in task (c), the context is low in each time-step and the model must capture dynamics to make best use of the context seen thus far. In Fig 3, we illustrate a sample of validation set in task (c) with predictions from ASNP and the baselines. As shown, even for the points not shown recently, ASNP predicts better than the baselines with lower uncertainty. ANP and SNP fail to predict those points well. This shows that the update of the imaginary queries and the representations is necessary over and above the sequential update of the global representations or the expensive modeling of the entire past using the time information. More qualitative results are in Appendix E.1.

2D regression: We take a step further to evaluate our model on a more challenging 2D setting where the queries are the pixel locations and the regression is over the pixel values of images. The image is a 42×42-sized black canvas with a moving MNIST digit (sized 28×28) or a cropped CelebA face (resized to 32×32). The digits or the faces start moving from a random location heading towards a direction randomly selected at $t = 0$ with a speed of 3 pixels per time-step and bouncing perfectly upon encountering a wall. Like in the regression task, a small Gaussian noise is added to the location obtained after each transition. Same as the regression task, 3 context regimes are designed. In regime (a) and (b), n is either 0 or $n \in [5, 500]$ while m is drawn from $[1, 501 - n]$. In (c), n is either 0 or 30 and m is drawn from $[1, 51 - n]$. The target set subsumes the context.

We present the quantitative results for 2D regression tasks in Fig. 4. From these, we infer the following points. We observe that SNP performs better than ANP except in the early time-steps of

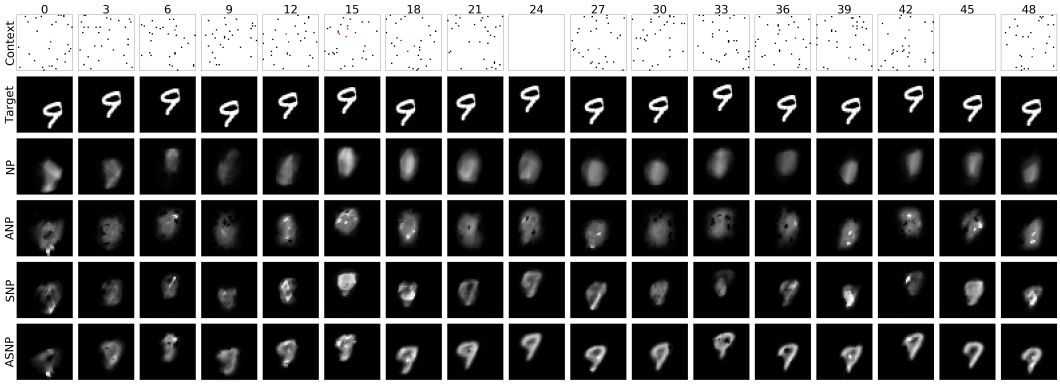

Figure 5: 2D moving MNIST regression samples from scenario (c). White pixels at contexts (first rows) are drawn as red to visualization.

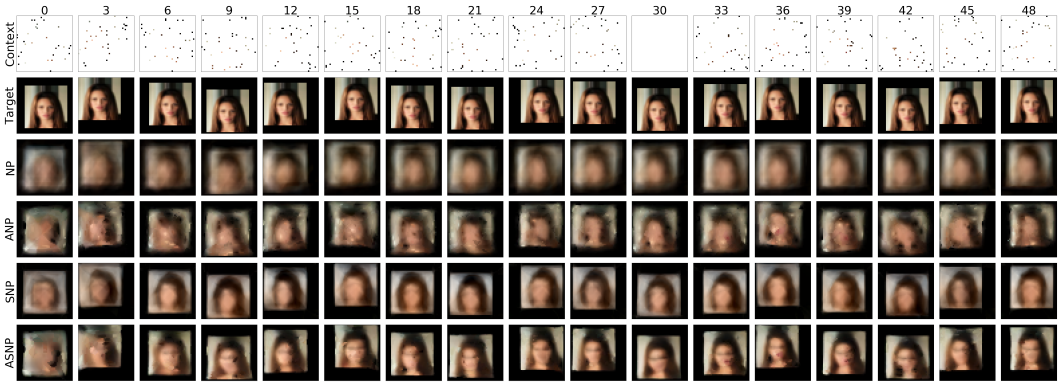

Figure 6: 2D moving CelebA regression samples from scenario (c).

scenario (a) when a large number of contexts points are shown without interruptions. We reason that in the presence of such ample context points, NP and SNP still suffer from under-fitting while ANP can make an effective use of those points without having to model the dynamics. But when the task setting demands that the model capture the dynamics, as in task (c), both NP and ANP perform poorer than SNP and ASNP. This is more pronounced in the 2D setting than in the 1D because tracking dynamics of images without explicit temporal modeling is harder. We further note that MNIST digits are simpler than the CelebA faces. Therefore in the latter case, SNP struggles with underfitting and ANP struggles with modeling the dynamics, causing them both to furnish similar performances in task (b). ASNP does not suffer from either of these problems and it outperforms all baselines in all the tasks. In scenario (a), we also note a sudden performance drop in NP, ANP and ASNP. While unsurprising in NP and ANP, it also occurs in ASNP due to sudden switch-over to a purely imaginary context. This drop does not appear in the 1D tasks since the number of context points provided are fewer and therefore ASNP is less impacted by their absence.

We also evaluate the effect of varying the imaginary context sizes on the moving MNIST data set for the scenarios (a) and (b). We observe that a larger k yields better performance. This gain, however, saturates quickly as the improvement between $k = 9$ and $k = 100$ is not large. This shows that the imaginary context can represent the stochastic process with only a few points.

In Fig. 5 and Fig. 6, we present the qualitative results on the two data sets for scenario (c). Here, we note that NP and ANP can produce the digit location but not the appearance. SNP captures the appearance of the digit but not clearly. Although ASNP also produces fuzzy appearance in the early time-steps, the digit becomes identifiable by $t = 10$ and the details keep getting clearer over $t = 20$. We also see a similar trend for the moving CelebA task. More illustrations of scenarios (a), (b) and (c) are included in Appendix E.2.

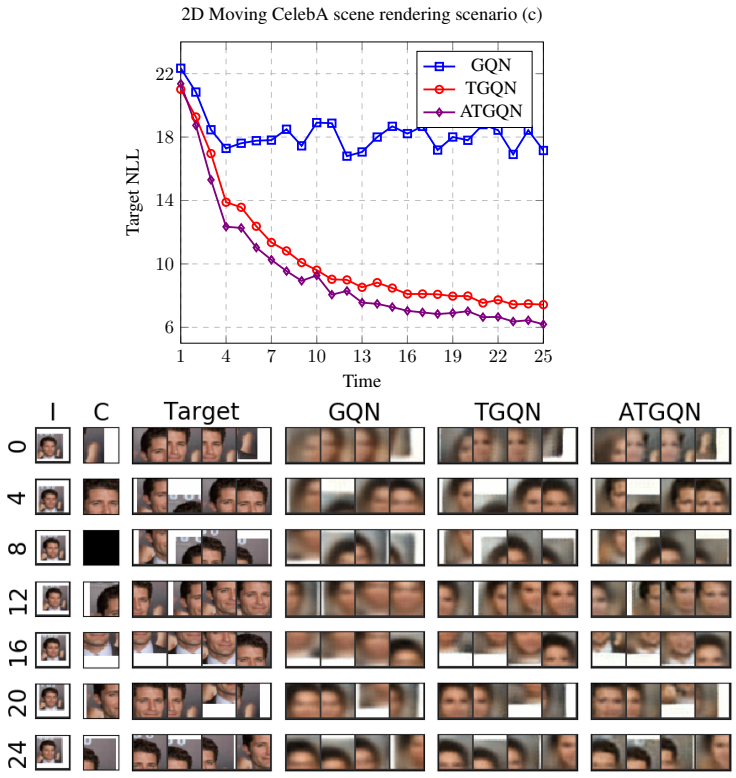

Figure 7: Top: Target NLL on moving CelebA 2D scene rendering task for scenario (c) and **Bottom:** examples.

2D scene rendering: We also test ASNP with the application of ASNP to a GQN architecture on the moving CelebA rendering task. We call it Attentive TGQN (ATGQN). To design this, we use the Temporal-ConvDRAW as in TGQN (Singh et al., 2019) for encoding a global latent. For computing a query-dependent representation, we apply the ASNP's attention encoder with imagination to get a scene-wise matrix. We use the same decoder as in TGQN. While it is a limited solution working on the task where y_t can be divided as points with queries, it resolves the underfitting for overlapping pixels between the context and the target. Unlike NPs, partial overlap exists, which is difficult to predict by scene-wise attention. More details about the model architecture is described in Appendix B.2.

The canvas size of the data set is 80×80 and cropped face image size is resized in 64×64. The direction of motion and initial position are randomly selected at $t = 0$ with 13 pixels per time-step speed. The size of output y is 32×32. The sequence length is 25. One context view is provided to the model at each of the randomly chosen 20 time-steps out of 25 time-steps. and an empty set is given on the remaining 5 time-steps. The target size m is drawn in the range $[1, 11 - n]$.

In Fig. 7, we show quantitative and qualitative results for scenario (c). We can see that ATGQN outperforms TGQN because for the overlapping areas, ATGQN resolves the underfitting. The performance gap between TGQN and ATGQN, however, is smaller compared to the previously seen 1D and 2D regression tasks. A similar trend is observed in scenario (a) whose results are in Appendix D.2. The reason for this is low uncertainty in the presence of the context that provides a wide view of the canvas. This and the small number of context causes lesser underfitting than what would have arisen had the context observations been highly partial as in the regression tasks. More examples are included in Appendix E.3.

6 Conclusion

In this paper, we addressed the problem of underfitting that plagues Neural Processes and Sequential Neural Processes. Although this is resolved in the former by ANP, it is not possible to resolve it in

SNP with direct application of attention since SNP cannot store the past context. We introduced Attentive Sequential Neural Processes which comprises a novel memory mechanism of imaginary context to resolve the underfitting. It not only compresses the past knowledge but does it in a way that is geared towards making good predictions with low uncertainty. In the experiments, the proposed model shows superior performance on various tasks for a number of sequential scenarios.

Since ASNP is a model based on attention, its limitations include those of any attention-based model. One such case is the scene-wise attention. Scene-wise attention cannot fit properly for partial overlapped region between context and target. Although we partially solve it with pixel-wise attention, it would still be interesting to encode query dependent representation for scene observation.

REFERENCES

Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. Tensorflow: A system for large-scale machine learning. In *12th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 16)*, pp. 265–283, 2016.

Howard Eichenbaum. Memory: organization and control. *Annual review of psychology*, 68:19–45, 2017.

SM Ali Eslami, Danilo Jimenez Rezende, Frederic Besse, Fabio Viola, Ari S Morcos, Marta Garnelo, Avraham Ruderman, Andrei A Rusu, Ivo Danihelka, Karol Gregor, et al. Neural scene representation and rendering. *Science*, 360(6394):1204–1210, 2018.

Marta Garnelo, Dan Rosenbaum, Chris J Maddison, Tiago Ramalho, David Saxton, Murray Shanahan, Yee Whye Teh, Danilo J Rezende, and SM Eslami. Conditional neural processes. *arXiv preprint arXiv:1807.01613*, 2018a.

Marta Garnelo, Jonathan Schwarz, Dan Rosenbaum, Fabio Viola, Danilo J Rezende, SM Eslami, and Yee Whye Teh. Neural processes. *arXiv preprint arXiv:1807.01622*, 2018b.

Karol Gregor, Ivo Danihelka, Alex Graves, Danilo Jimenez Rezende, and Daan Wierstra. Draw: A recurrent neural network for image generation. *arXiv preprint arXiv:1502.04623*, 2015.

Karol Gregor, Frederic Besse, Danilo Jimenez Rezende, Ivo Danihelka, and Daan Wierstra. Towards conceptual compression. In *Advances In Neural Information Processing Systems*, pp. 3549–3557, 2016.

Hyunjik Kim, Andriy Mnih, Jonathan Schwarz, Marta Garnelo, Ali Eslami, Dan Rosenbaum, Oriol Vinyals, and Yee Whye Teh. Attentive neural processes. *arXiv preprint arXiv:1901.05761*, 2019a.

Hyunjik Kim, Andriy Mnih, Jonathan Schwarz, Marta Garnelo, Ali Eslami, Dan Rosenbaum, Oriol Vinyals, and Yee Whye Teh. Attentive neural processes. *arXiv preprint arXiv:1901.05761*, 2019b.

Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

Ananya Kumar, SM Eslami, Danilo J Rezende, Marta Garnelo, Fabio Viola, Edward Lockhart, and Murray Shanahan. Consistent generative query networks. *arXiv preprint arXiv:1807.02033*, 2018.

Yann LeCun, Léon Bottou, Yoshua Bengio, Patrick Haffner, et al. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Juho Lee, Yoonho Lee, Jungtaek Kim, Adam Kosiorek, Seungjin Choi, and Yee Whye Teh. Set transformer: A framework for attention-based permutation-invariant neural networks. In *International Conference on Machine Learning*, pp. 3744–3753, 2019.

Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *Proceedings of the IEEE international conference on computer vision*, pp. 3730–3738, 2015.

Christos Louizos, Xiahan Shi, Klamer Schutte, and Max Welling. The functional neural process. *arXiv preprint arXiv:1906.08324*, 2019.

Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th international conference on machine learning (ICML-10)*, pp. 807–814, 2010.

Alexander Pritzel, Benigno Uria, Sriram Srinivasan, Adria Puigdomenech Badia, Oriol Vinyals, Demis Hassabis, Daan Wierstra, and Charles Blundell. Neural episodic control. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pp. 2827–2836. JMLR. org, 2017.

Dan Rosenbaum, Frederic Besse, Fabio Viola, Danilo J Rezende, and SM Eslami. Learning models for visual 3d localization with implicit mapping. *arXiv preprint arXiv:1807.03149*, 2018.

Daniel L Schacter. Constructive memory: past and future. *Dialogues in clinical neuroscience*, 14 (1):7, 2012.

Gautam Singh, Jaesik Yoon, Youngsung Son, and Sungjin Ahn. Sequential neural processes. *arXiv preprint arXiv:1906.10264*, 2019.

Edward Snelson and Zoubin Ghahramani. Sparse gaussian processes using pseudo-inputs. In *Advances in neural information processing systems*, pp. 1257–1264, 2006.

Michalis Titsias. Variational learning of inducing variables in sparse gaussian processes. In *Artificial Intelligence and Statistics*, pp. 567–574, 2009.

Jakub M Tomczak and Max Welling. Vae with a vampprior. *arXiv preprint arXiv:1705.07120*, 2017.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pp. 5998–6008, 2017.

SHI Xingjian, Zhourong Chen, Hao Wang, Dit-Yan Yeung, Wai-Kin Wong, and Wang-chun Woo. Convolutional lstm network: A machine learning approach for precipitation nowcasting. In *Advances in neural information processing systems*, pp. 802–810, 2015.

APPENDIX A ELBO DERIVATIONS

$$
\begin{aligned}
&\log P(Y|X,C) \\
&= \log \mathbb{E}_{Q(Z,C^i|C^r,D)} \left[\frac{P(Y,Z,C^i|X,C^r)}{Q(z,C^i|C^r,D)} \right] \\
&= \log \mathbb{E}_{Q(Z,C^i|C^r,D)} \left[\prod_{t=1}^{T} \frac{P(y_t|x_t,z_t,C_t)P(z_t|z_{<t},C_t)P(U_t^i|X_t^i,C_{<t},C_t^r)P(X_t^i|C_{<t},C_t^r)}{Q(z_t|z_{<t},C_t^i,C^r,D)Q(U_t^i|X_t^i,C_{<t}^i,C^r,D)Q(X_t^i|C_{<t}^i,C^r,D)} \right] \\
&\geq \mathbb{E}_{Q(Z,C^i|C^r,D)} \left[\log \prod_{t=1}^{T} \frac{P(y_t|x_t,z_t,C_t)P(z_t|z_{<t},C_t)P(U_t^i|X_t^i,C_{<t},C_t^r)P(X_t^i|C_{<t},C_t^r)}{Q(z_t|z_{<t},C_t^i,C^r,D)Q(U_t^i|X_t^i,C_{<t}^i,C^r,D)Q(X_t^i|C_{<t}^i,C^r,D)} \right] \\
&= \mathbb{E}_{Q(Z,C^i|C^r,D)} \sum_{t=1}^{T} \left[\log \frac{P(y_t|x_t,z_t,C_t)P(z_t|z_{<t},C_t)P(U_t^i|X_t^i,C_{<t},C_t^r)P(X_t^i|C_{<t},C_t^r)}{Q(z_t|z_{<t},C_t^i,C^r,D)Q(U_t^i|X_t^i,C_{<t}^i,C^r,D)Q(X_t^i|C_{<t}^i,C^r,D)} \right] \\
&= \mathbb{E}_{Q(Z,C^i|C^r,D)} \sum_{t=1}^{T} [\log P(y_t|x_t,z_t,C_t) - \log \frac{Q(z_t|z_{<t},C_t^i,C^r,D)}{P(z_t|z_{<t},C_t)} - \log \frac{Q(U_t^i|X_t^i,C_{<t}^i,C^r,D)}{P(U_t^i|X_t^i,C_{<t},C_t^r)} \\
&- \log \frac{Q(X_t^i|C_{<t}^i,C^r,D)}{P(X_t^i|C_{<t},C_t^r)}] \\
&= \sum_{t=1}^{T} \mathbb{E}_{\prod_{t'=1}^{t} Q(z_{t'}|z_{<t'},C_{t'}^i,C^r,D)Q(C_{t'}^i|C_{<t'}^i,C^r,D)} [\log P(y_t|x_t,z_t,C_t)] \\
&- \mathbb{E}_{\prod_{t'=1}^{t} Q(z_{t'}|z_{<t'},C_{t'}^i,C^r,D)Q(C_{t'}^i|C_{<t'}^i,C^r,D)} \left[\log \frac{Q(z_t|z_{<t},C_t^i,C^r,D)}{P(z_t|z_{<t},C_t)} \right] \\
&- \mathbb{E}_{\prod_{t'=1}^{t} Q(U_{t'}^i|X_{t'}^i,C_{<t'}^i,C^r,D)Q(X_{t'}^i|C_{<t'}^i,C^r,D)} \left[\log \frac{Q(U_t^i|X_t^i,C_{<t}^i,C^r,D)}{P(U_t^i|X_t^i,C_{<t},C_t^r)} \right] \\
&- \mathbb{E}_{\prod_{t'=1}^{t} Q(X_{t'}^i|C_{<t'}^i,C^r,D)} \left[\log \frac{Q(X_t^i|C_{<t}^i,C^r,D)}{P(X_t^i|C_{<t},C_t^r)} \right] \\
&= \sum_{t=1}^{T} \mathbb{E}_{\prod_{t'=1}^{t} Q(z_{t'}|z_{<t'},C_{t'}^i,C^r,D)Q(C_{t'}^i|C_{<t'}^i,C^r,D)} [\log P(y_t|x_t,z_t,C_t)] \\
&- \mathbb{E}_{Q(C_t^i|C_{<t}^i,C^r,D)\prod_{t'=1}^{t-1} Q(z_{t'}|z_{<t'},C_{t'}^i,C^r,D)Q(C_{t'}^i|C_{<t'}^i,C^r,D)} \mathbb{KL}(Q(z_t|z_{<t},C_t^i,C^r,D) \parallel P(z_t|z_{<t},C_t)) \\
&- \mathbb{E}_{Q(X_t^i|C_{<t}^i,C^r,D)\prod_{t'=1}^{t-1} Q(U_{t'}^i|X_{t'}^i,C_{<t'}^i,C^r,D)Q(X_{t'}^i|C_{<t'}^i,C^r,D)} \mathbb{KL}(Q(U_t^i|X_t^i,C_{<t}^i,C^r,D) \parallel P(U_t^i|X_t^i,C_{<t},C_t^r)) \\
&- \mathbb{E}_{\prod_{t'=1}^{t-1} Q(X_{t'}^i|C_{<t'}^i,C^r,D)} \mathbb{KL}(Q(X_t^i|C_{<t}^i,C^r,D) \parallel P(X_t^i|C_{<t},C_t^r)) \\
&= \sum_{t=1}^{T} \mathbb{E}_{\prod_{t'=1}^{t} Q(z_{t'}|z_{<t'},C_{t'}^i,C^r,D)Q(C_{t'}^i|C_{<t'}^i,C^r,D)} [\log P(y_t|x_t,z_t,C_t)] \\
&- \mathbb{E}_{Q(C_t^i|C_{<t}^i,C^r,D)\prod_{t'=1}^{t-1} Q(z_{t'}|z_{<t'},C_{t'}^i,C^r,D)Q(C_{t'}^i|C_{<t'}^i,C^r,D)} \mathbb{KL}(Q(z_t|z_{<t},C_t^i,C^r,D) \parallel P(z_t|z_{<t},C_t)) \\
&- \mathbb{E}_{\prod_{t'=1}^{t-1} Q(C_{t'}^i|C_{<t'}^i,C^r,D)} \mathbb{KL}(Q(C_t^i|C_{<t}^i,C^r,D) \parallel P(C_t^i|C_{<t}^i,C_t^r)) \\
&= \sum_{t=1}^{T} \mathbb{E}_{\prod_{t'=1}^{t} Q(z_{t'}|z_{<t'},C_{t'}^i,C^r,D)Q(C_{t'}^i|C_{<t'}^i,C^r,D)} [\log P(y_t|x_t,z_t,C_t)] \\
&- \mathbb{E}_{\prod_{t'=1}^{t-1} Q(z_{t'},C_t^{i\prime}|z_{<t'},C_{<t'}^i,C^r,D)} \mathbb{KL}(Q(z_t,C_t^i|z_{<t},C_{<t}^i,C^r,D) \parallel P(z_t,C_t^i|z_{<t},C_{<t}^i,C_t^r)) \\
&= \sum_{t=1}^{T} \mathbb{E}_{Q_\phi(z_t,C_t^i|C^r,D)} \left[\log P_\theta(y_t|x_t,z_t,C_t^i,C_t^r) \right] \\
&- \mathbb{E}_{Q_\phi(z_{<t},C_{<t}^i)} \left[\mathbb{KL} \left(Q_\phi(z_t,C_t^i|C^r,D) \parallel P_\theta(z_t,C_t^i|z_{<t},C_{<t}^i,C_t^r) \right) \right]
\end{aligned}
$$

where $Q_\phi(z_t, C_t^i|C^r, D) = \prod_{t'=1}^{t} Q(z_{t'}|z_{<t'}, C_{t'}^i, C^r, D)Q(C_{t'}^i|C_{<t'}^i, C^r, D)$ and KL expectation $Q_\phi(z_{<t}, C_{<t}^i) = \prod_{t'=1}^{t-1} Q(z_{t'}, C_t^{i'}|z_{<t'}, C_{<t'}^i, C^r, D)$ to simplicity.

APPENDIX B MODEL DETAILS

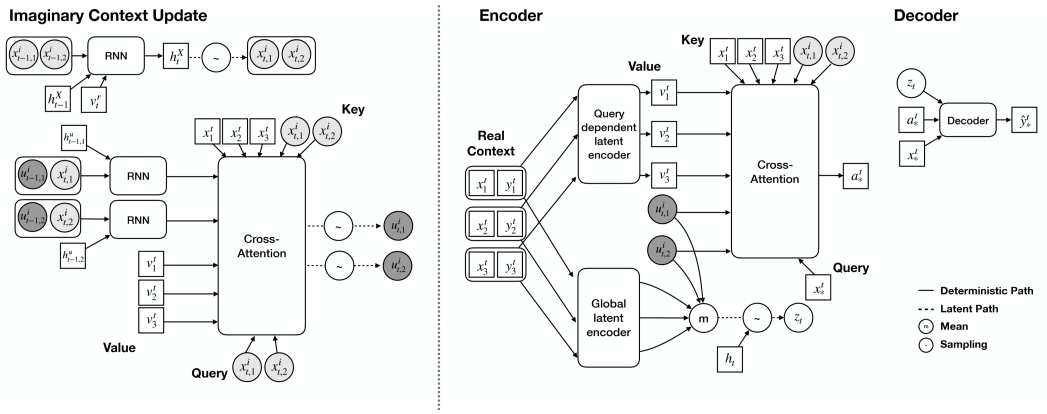

Figure 8: Model architecture of Attentive Sequential Neural Processes (ASNP) imaginary context update (**left**) and encoder/decoder (**right**)

B.1 NPS

Every models share a basic architecture (e.g. context encoder and decoder). They have two encoders. One of them is for a global latent. It consists of 3 layers MLP with ReLU (Nair & Hinton, 2010) is used as encoder and 2 layers MLP is used to encode and sample a latent. Another encoder is consisted of 6 layers MLP with ReLU activation function and mean function for NP and SNP and attention module for ANP and ASNP. Decoder is consisted of 4 layers MLP with ReLU activation function and 1 layer MLP is used to sample $\hat{y}$ with uncertainty.

For non sequential models (NP and ANP), time is encoded as a normalized float scalar, $e_t = 0.25 + 0.5(t/T)$, where T is the length of sequence of data. After encoding, it is appended in query as $x' = (x, e_t)$.

For attention models (ANP and ASNP), Multihead attention (Vaswani et al., 2017) is used because it showed the best performance in a variety of attention methods (Kim et al., 2019b). ASNP has two attention modules, which shares parameters.

For sequential models (SNP and ASNP), they uses same temporal architecture for a global latent. LSTM with a default setting of TensorFlow (Abadi et al., 2016) is used and the dimension of hidden unit of LSTM is d_r where d_r is the dimension of representations. For ASNP, two RNN modules for the imaginary queries and representations is LSTM with a default setting of Tensorflow. The dimension of hidden unit of LSTMs is d_r. Here, the number of state of LSTM for representations is $k\times$batch size.

The dimension of representation d_r is 128 and the dimension of query d_q is 1 for 1D regression and 2 for 2D regression. Note that for non-sequential model, d_q is 2 and 3 by attaching encoded time e_t. The number of imaginary context is 25. The initial imaginary context C_0^i is trainable parameters. Learning rate is 0.0001 and batch size is 16 for 1D regression and 8 for 2D regression. The diagram for ASNP architecture is in Fig. 8.

B.2 GQNS

For an encoder, tower representation (Eslami et al., 2018) is used. A decoder is similar between models (GQN, TGQN and ATGQN) based on the decoder of CGQN (Kumar et al., 2018) that is a deterministic generative model working on auto-regressive manner. The different is inputs that are described in bellows.

For GQN, time is encoded and appended in query same to NP and ANP. To sample z_t, we apply a convolutional DRAW(ConvDRAW) (Gregor et al., 2015; 2016) like GQN (Kumar et al., 2018). To render $\hat{y}$, the decoder inputs z_t, X_t and v_t^r that is an output of the decoder. v_t^r is averaged representation that is an input for every target queries.

For TGQN, Temporal-ConvDRAW (Singh et al., 2019) is applied to sample z_t with ConvLSTM (Xingjian et al., 2015) state, h_t. We take RSSM to implement as (Singh et al., 2019). The decoder input is a set of the global latent z_t, the target queries X_t and ConvLSTM state h_t, in which, z_t and h_t are independent to target queries.

For ATGQN, a global latent is sampled same to TGQN and a query dependent representation of ASNP is used. To make a pixel-wise path, we split scene view as pixel view with queries per each pixel. When many context are given, overlapped between context exists. We leave it to optimize batch operation. We encode the pixel-wise representation with 2 convolutional layers to make same dimension with scene-wise, the dimension of which is $[\text{scene-height}/4, \text{scene-height}/4, 3]$. The input of decoder are z_t, X_t and $r_t = \{r_i^t\}_{i \in \mathcal{I}(D)}$ that is query dependent representations for each target query x_i^t.

The dimension of representation d_r is 128. The number of steps for (Temporal) ConvDRAW to sample z_t is 3 and the dimension of z is 4. Note that z_t is the output of the last roll-out of (Temporal) ConvDRAW. For TGQN and ATGQN, the number of hidden unit for ConvLSTM is 40. The dimension of hidden unit on the decoder is 32 and the number of steps for auto-regressive decoding is 6 and output is cultivated. The query size is 2 (for GQN, it is 3 with e_t). The number of pseudo context is 100. Learning rate and batch size is 0.0001 and 4.

APPENDIX C GAUSSIAN PROCESS DATA SET

The kernel hyper-parameters, length-scale l and kernel-scale σ are selected randomly at $t = 0$ in $[0.7, 1.2]$ and $[1.0, 1.6]$ for scenarios (a) and (b). For scenario (c), l and σ are chosen in $[1.2, 1.9]$ and $[1.6, 3.1]$ at $t = 0$. The true underlying dynamics of the kernel hyper-parameters Δl and $\Delta\sigma$ are in $[-0.03, 0.03]$ and $[-0.05, 0.05]$ chosen at $t = 0$. Gaussian noise on the dynamics is $\sim \mathcal{N}(0, 0.1)$.

APPENDIX D ADDITIONAL EXPERIMENTAL RESULTS

D.1 LEARNING CURVE ON GP TASK

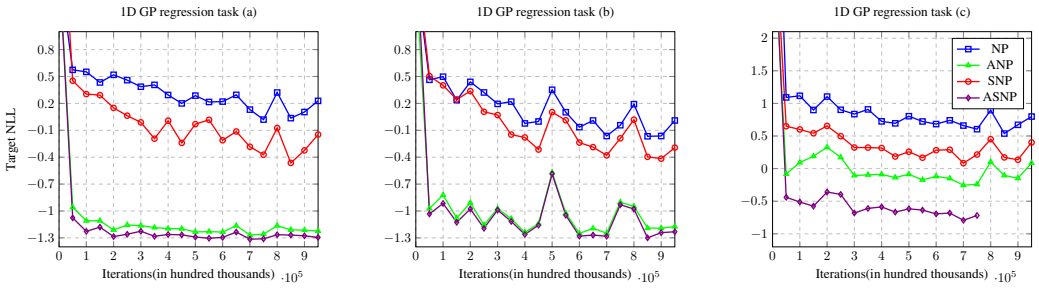

Figure 9: Target NLL for iteration.

In this section, we show target NLL for iterations (Figs. 9) and wall clock time (Figs. 10) for 1D GP data set. ASNP and ANP computation complexity is bigger than NP $\mathcal{O}(1)$ due to attention. ANP needs $\mathcal{O}(m \sum_t^T n_t)$ where n_t is the context size at time-step t. Because it needs to attention to entire context in current. ASNP computation complexity is $\mathcal{O}((n+k)m)$ for generating r_t and $\mathcal{O}((n+k)k)$ for updating the pseudo context. When the sequence of data T is not long, ANP computation complexity is smaller than the complexity of ASNP. However, Fig. 10 shows ASNP quickly saturates on lower loss in same time.

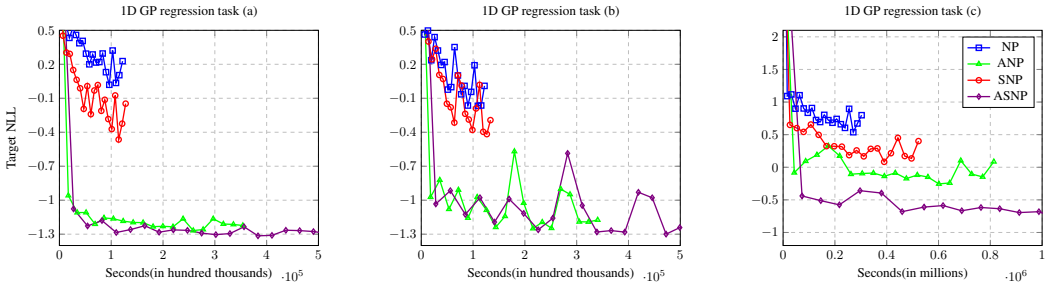

Figure 10: Target NLL for wall clock time.

D.2 2D RENDERING TASK FOR SCENARIO (A)

We also test our model on the moving CelebA rendering task for scenario (a). The sequence length T sets as 15 and the context is given to $t = 7$. The context size n is randomly selected in $[1, 10]$ or 0. The target size m is randomly chosen in $[1, 11 - n]$. Other environment setting is same to scenario (c).

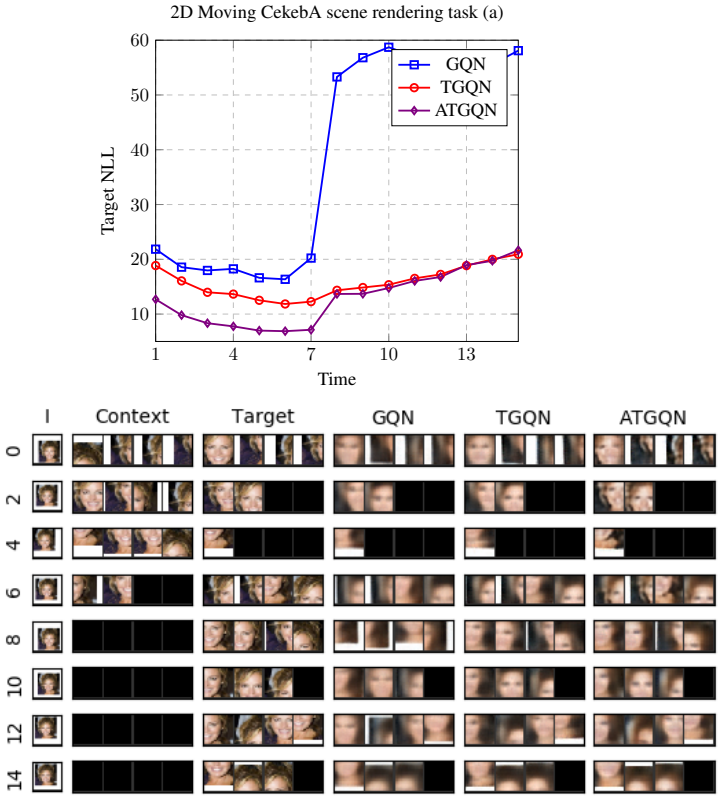

Figure 11: Top: Target NLL on moving CelebA 2D scene rendering task for scenario (a) and **Bottom:** examples.

Fig. 11 shows results that are both quantitative and qualitative. Target NLL value of GQN is very high. On the other hand, the generation performance is not the same as the target NLL. The explanation for this is inability to predict the complex face image's location. When context is given, ATGQN outperforms TGQN because ATGQN solves the overlapping region underfitting. ATGQN however shows similar outcomes to TGQN on non-context prediction. The explanation for this is less uncertainty due to a lot of contextual information.

APPENDIX E QUALITATIVE RESULTS

E.1 1D REGRESSION

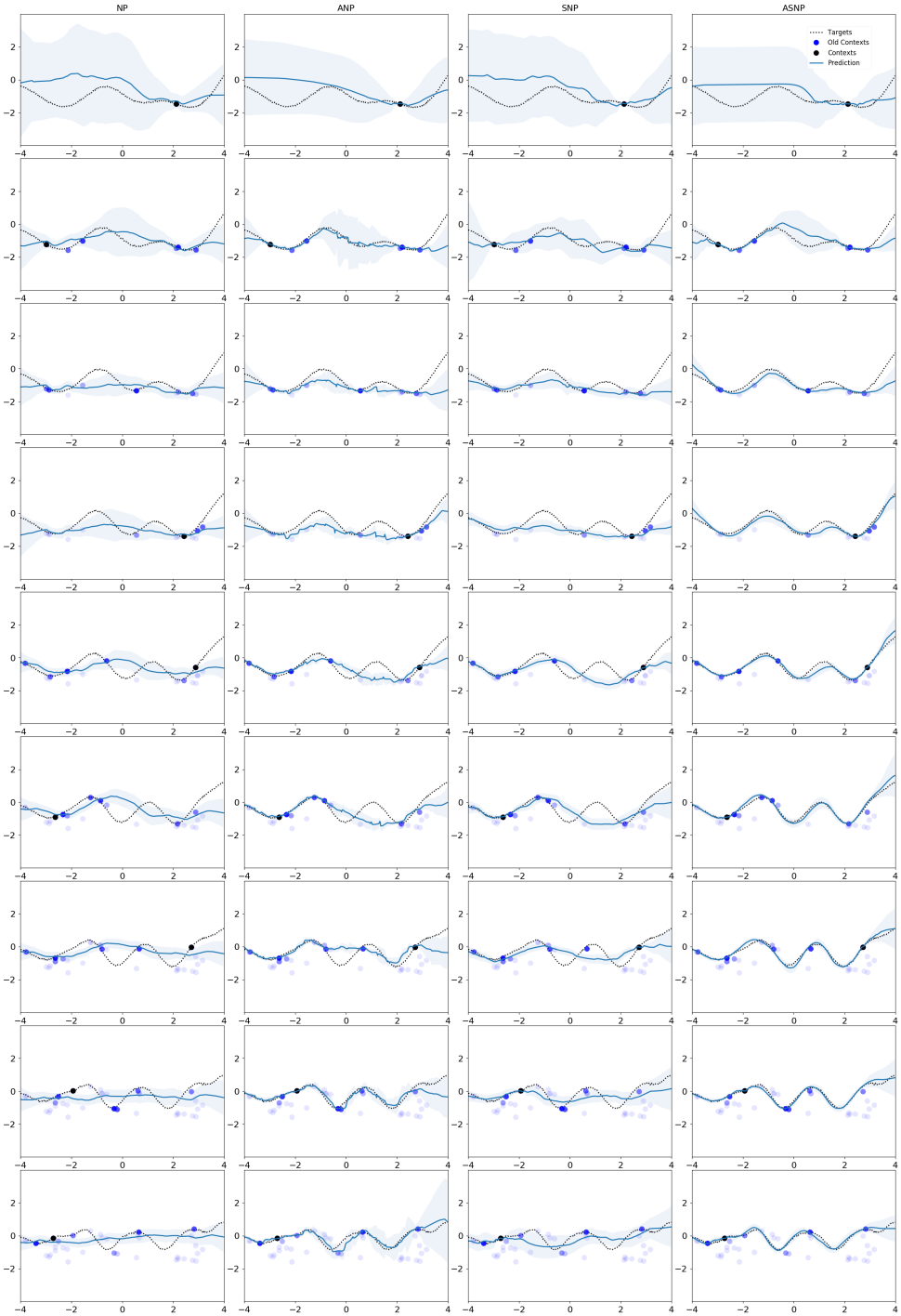

Figure 12: 1D GP regression examples for scenario (c). Columns are NP, ANP, SNP and ASNP. Each row is examples at each time-step. Due to space limitations, every 5th time-step is shown here instead of every time-step.

E.2 2D REGRESSION

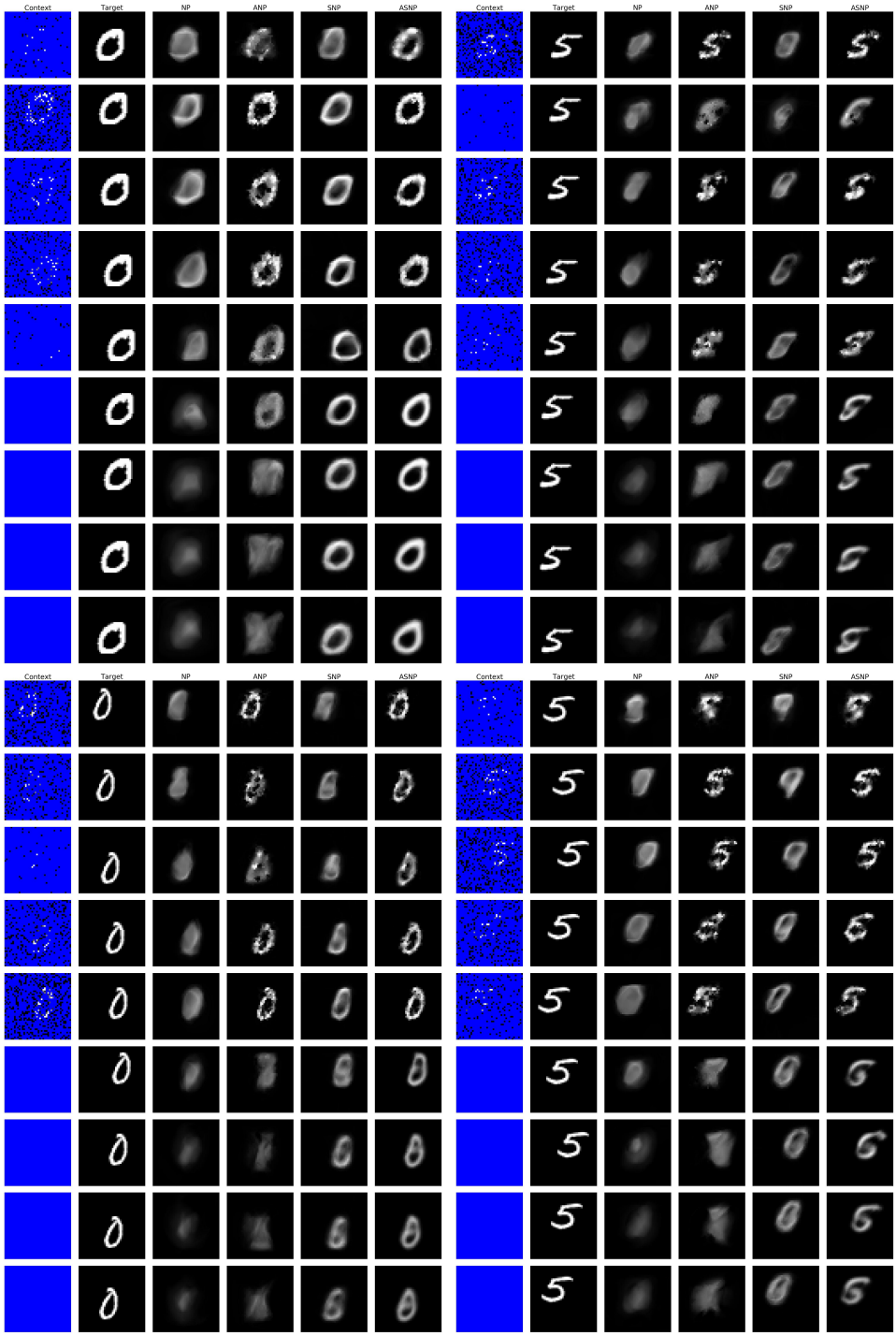

Figure 13: 2D moving MNIST regression examples for scenario (a). Columns are Context, Target, NP, ANP, SNP and ASNP. Each row is examples at each time-step. Due to space limitations, every 2th time-step is shown here instead of every time-step.

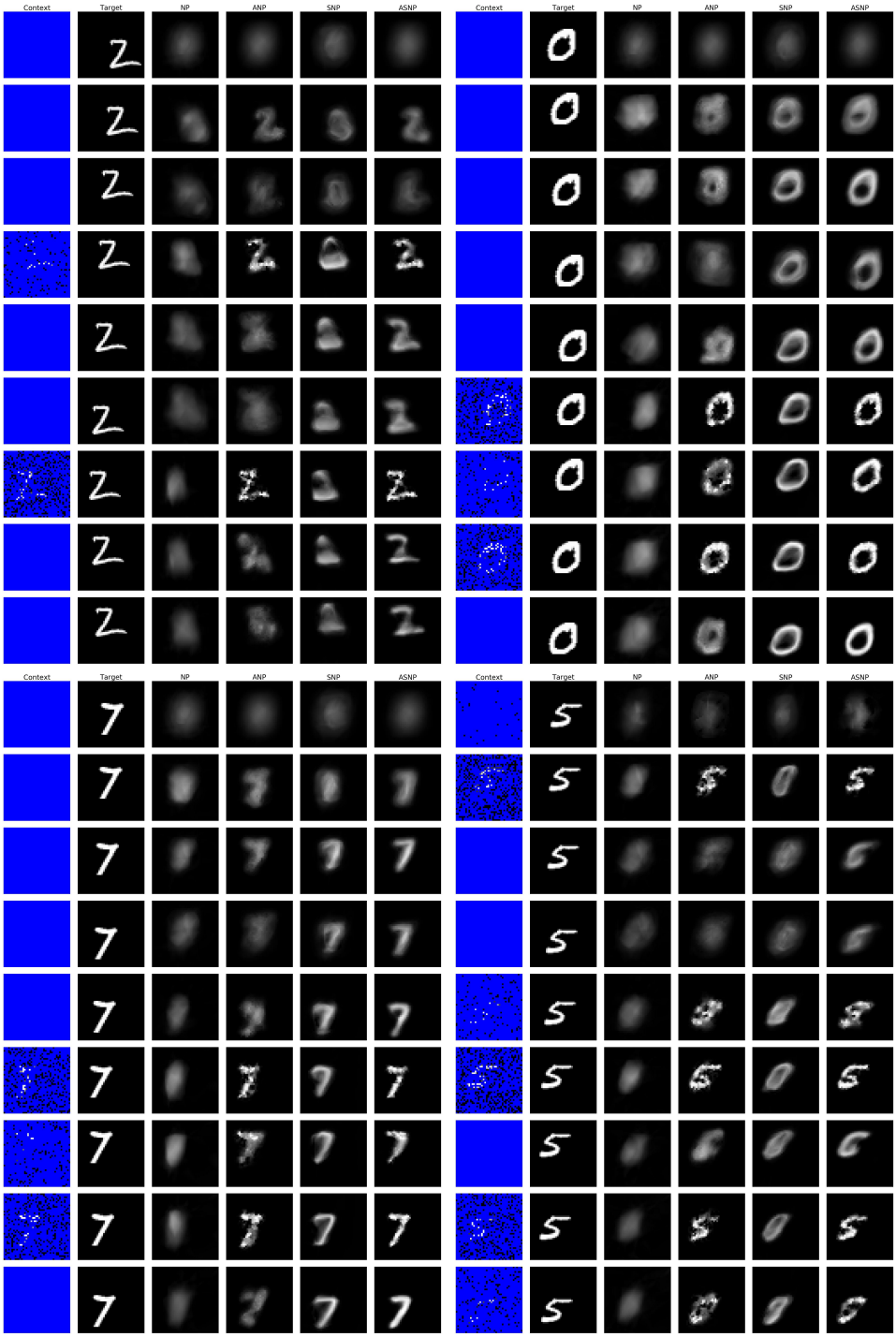

Figure 14: 2D moving MNIST regression examples for scenario (b). Columns are Context, Target, NP, ANP, SNP and ASNP. Each row is examples at each time-step. Due to space limitations, every 2th time-step is shown here instead of every time-step. It is shown as less than 10 time-steps have context because it doesn't show every time-steps.

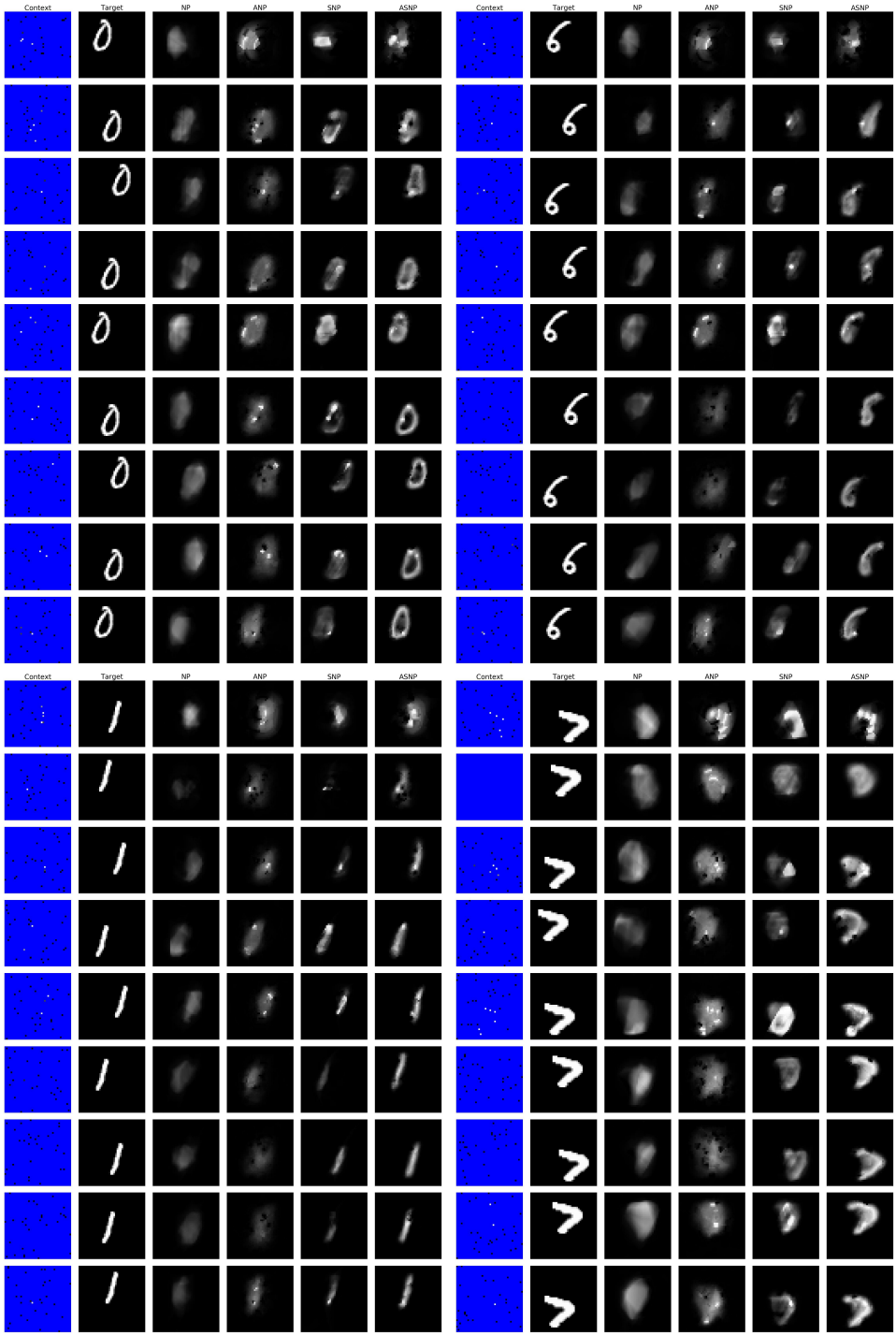

Figure 15: 2D moving MNIST regression examples for scenario (c). Columns are Context, Target, NP, ANP, SNP and ASNP. Each row is examples at each time-step. Due to space limitations, every 5th time-step is shown here instead of every time-step.

Figure 16: 2D moving CelebA regression examples for scenario (a). Columns are Context, Target, NP, ANP, SNP and ASNP. Each row is examples at each time-step. Due to space limitations, every 2th time-step is shown here instead of every time-step.

Figure 17: 2D moving CelebA regression examples for scenario (b). Columns are Context, Target, NP, ANP, SNP and ASNP. Each row is examples at each time-step. Due to space limitations, every 2th time-step is shown here instead of every time-step. It is shown as less than 10 time-steps have context because it doesn't show every time-steps.

Figure 18: 2D moving CelebA regression examples for scenario (c). Columns are Context, Target, NP, ANP, SNP and ASNP. Each row is examples at each time-step. Due to space limitations, every 5th time-step is shown here instead of every time-step.

E.3 2D SCENE RENDERING

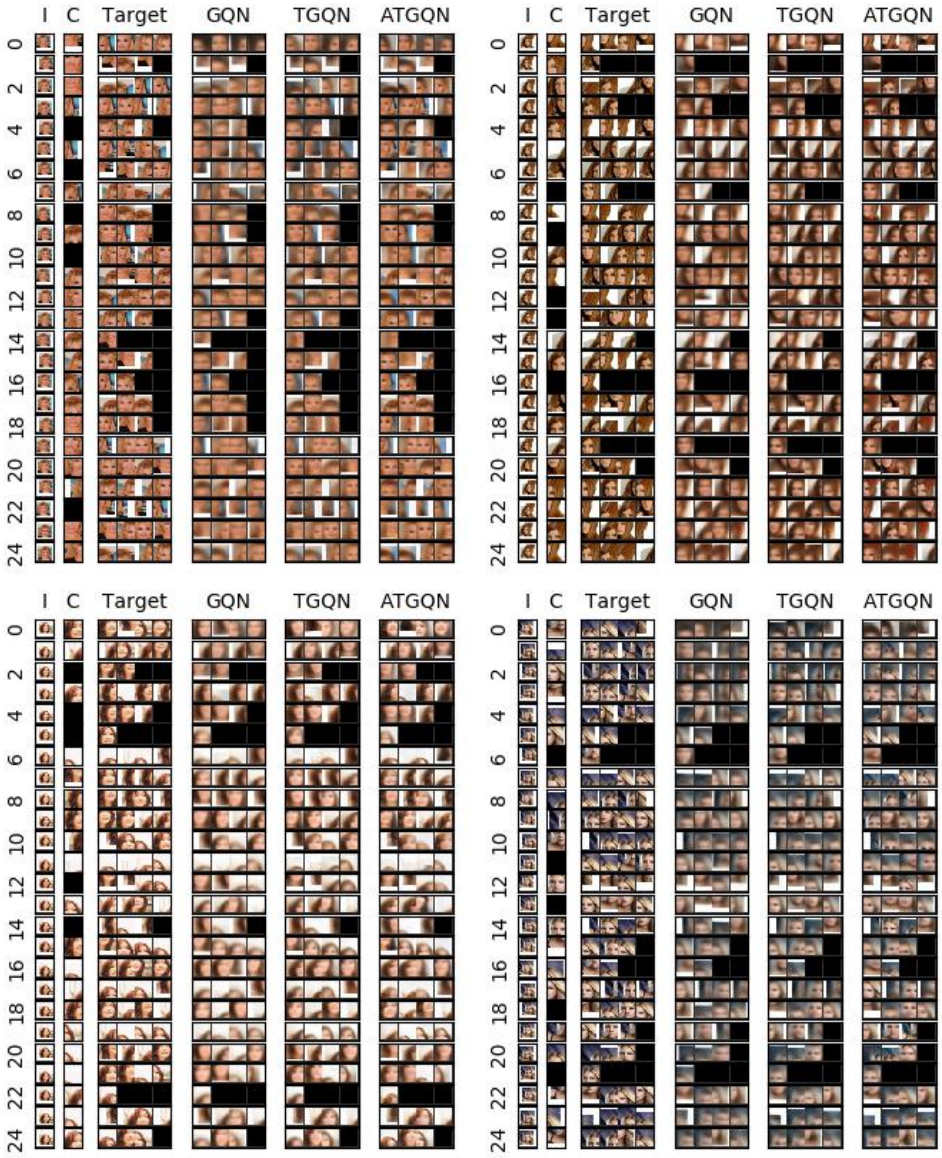

Figure 19: moving CelebA 2D scene rendering task examples for scenario (c). Columns are Image (I), Context (C), Target, GQN, TGQN, ATGQN. Each row is examples at each time-step.