# Learning to Decompose and Disentangle Representations for Video Prediction

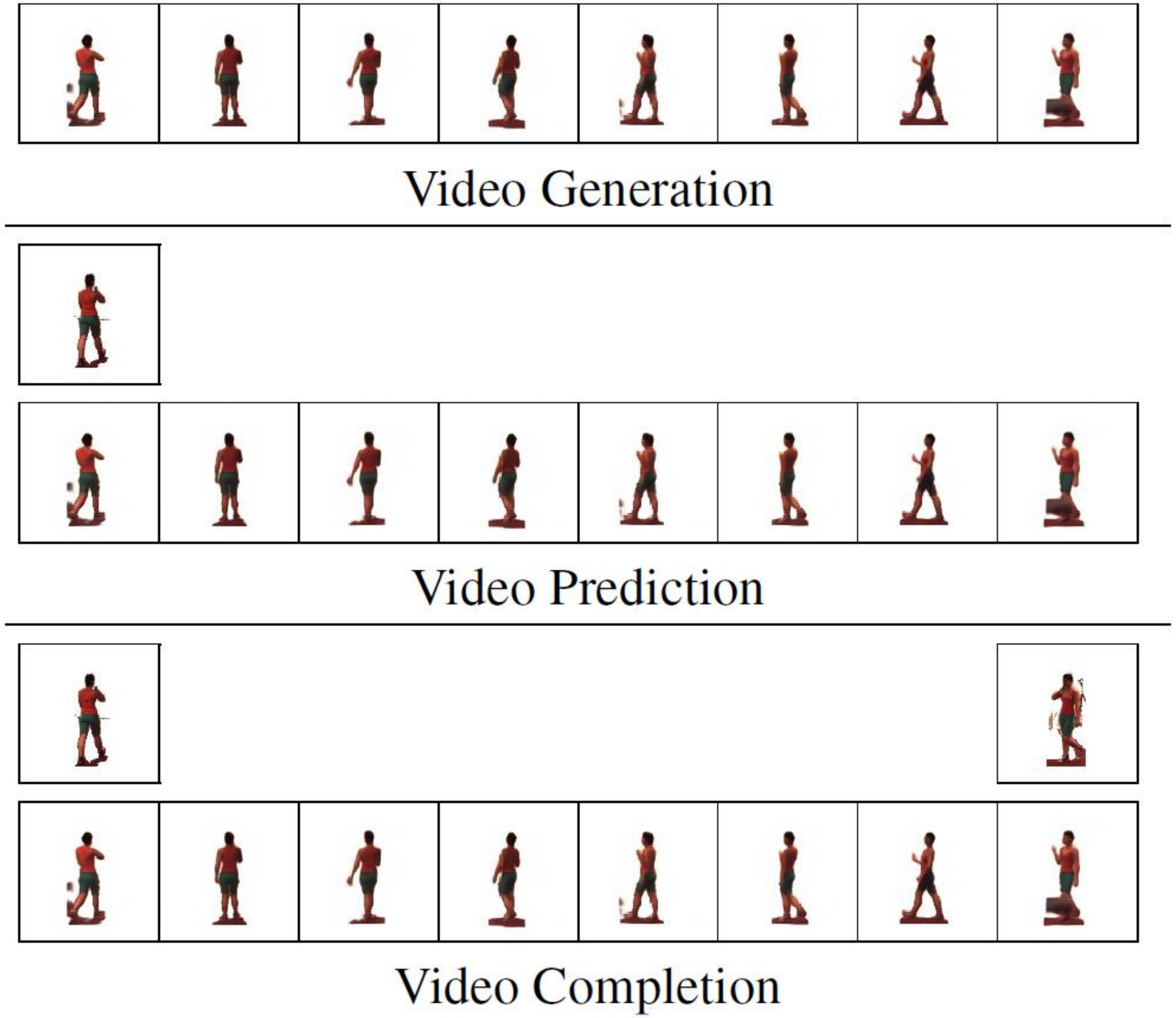
NIPS2018

2019.09.03

발표자 박성현

# 1 Introduction

## Video Prediction



# 1 Introduction

## Motivation

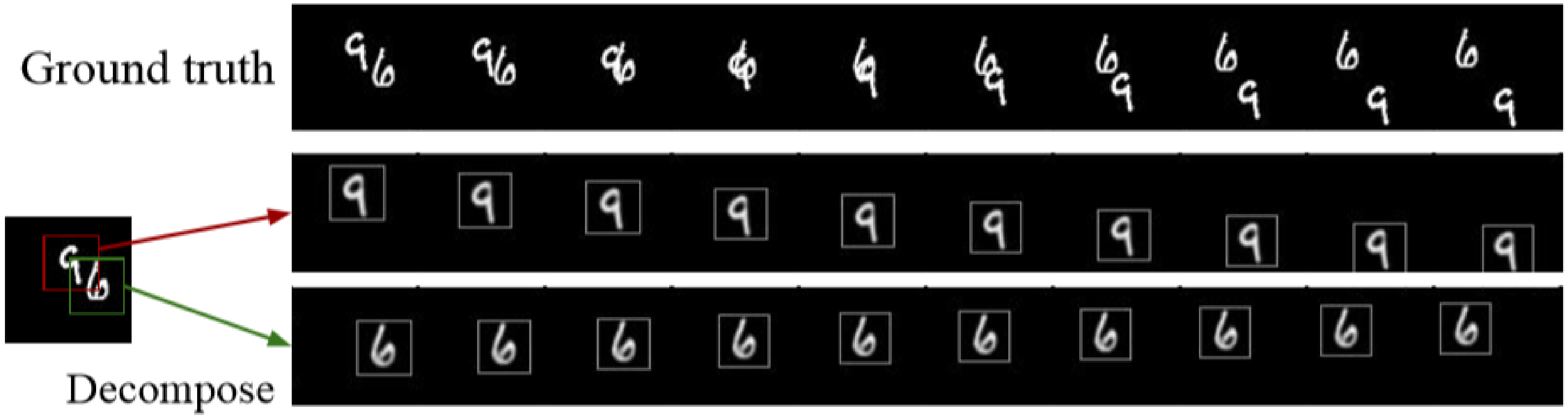



Figure 1: Our key insight is to decompose the video into several components. The prediction of each individual component is easier than directly predicting the whole image sequence. It is important to note that the decomposition is learned automatically without explicit supervision.

# 2 Model

## Model Overview

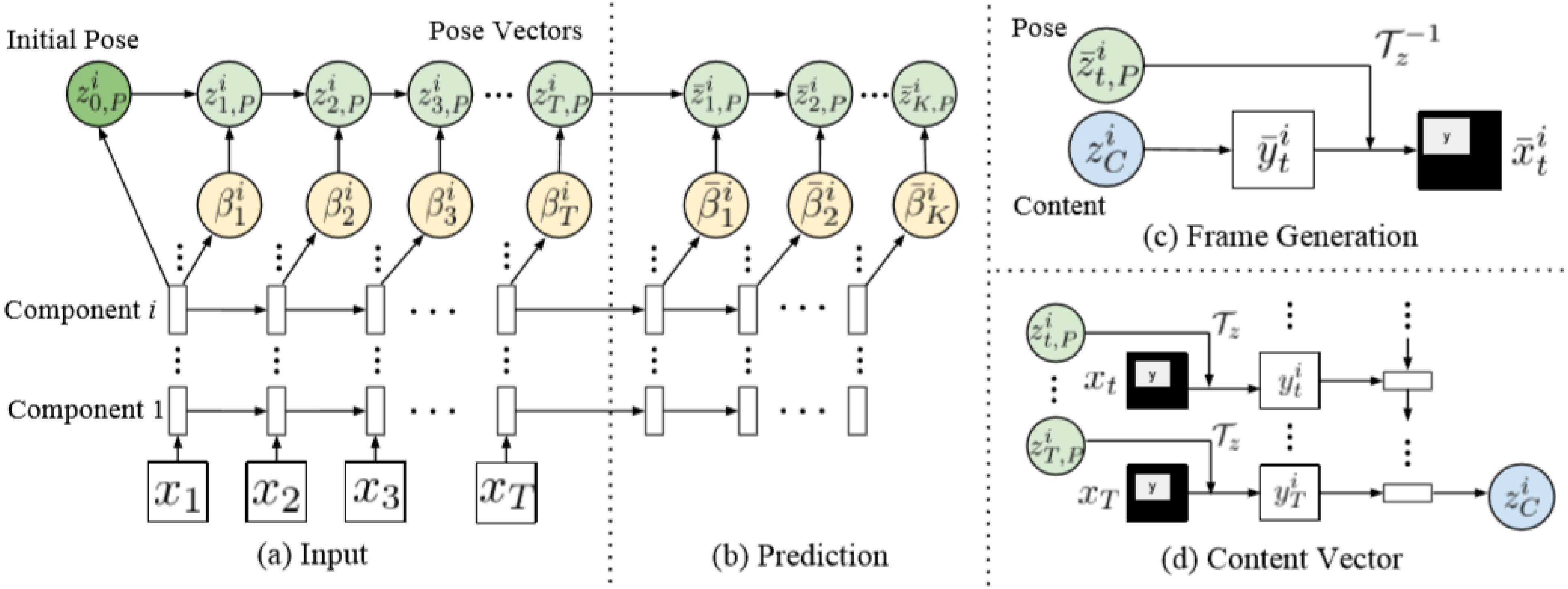



Figure 2: Overview of our model implementation. (a) We use 2D recurrence to implement $q(z^i_{1:T}|x_{1:T})$ to model both the temporal and dependency between components. (b) The prediction RNN is used only to predict the pose vector. (c) Our frame generation model generates different image with the same content using inverse spatial transformer. (d) A single content vector $z^i_C$ is obtained for each component from input $x_{1:T}$ and pose vectors $z^i_{1:T}$.

# 2 Model

## Decompositional Disentangled Predictive Auto-Encoder (DDPAE)

**[Video Prediction Problem]**

Frame Decoder

Temporal Encoder

$$p(\bar{x}_{1:K}|x_{1:T}) = \iint p(\bar{x}_{1:K}|\bar{z}_{1:K})p(\bar{z}_{1:K}|z_{1:T})p(z_{1:T}|x_{1:T})\, d\bar{z}_{1:K}\, dz_{1:T}$$

Prediction Model

**[Decomposition]**

각 Component로 Decompose

$$\bar{x}_{1:K} = \sum_{i=1}^{N} \bar{x}^i_{1:K}, \quad x_{1:T} = \sum_{i=1}^{N} x^i_{1:T}$$

$$p(\bar{x}^i_{1:K}|x^i_{1:T}) = \iint p(\bar{x}^i_{1:K}|\bar{z}^i_{1:K})p(\bar{z}^i_{1:K}|z^i_{1:T})p(z^i_{1:T}|x^i_{1:T})\, d\bar{z}^i_{1:K}\, dz^i_{1:T}$$

**[Disentangle]**

Prediction is reduced to just predicting the low-dimensional pose vectors

$$p(\bar{z}^i_{1:K}|z^i_{1:T}) = p(\bar{z}^i_{1:K,P}|z^i_{1:T,P}), \quad \bar{z}^i_t = [z^i_C, \bar{z}^i_{t,P}], \quad z^i_t = [z^i_C, z^i_{t,P}]$$

Content Vector

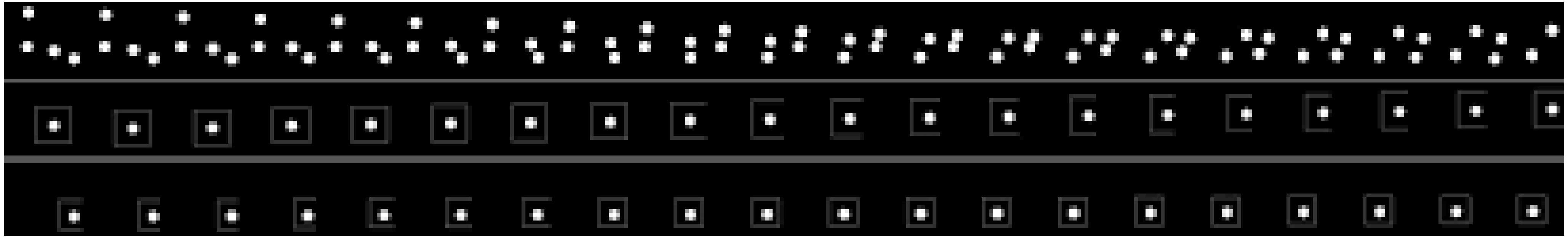
Pose Vector

# 3 Experiments

## Datasets

[Moving MNIST]

[Bouncing Balls]



# 3 Experiments

## Evaluating Decompositional Disentangled Video Representation

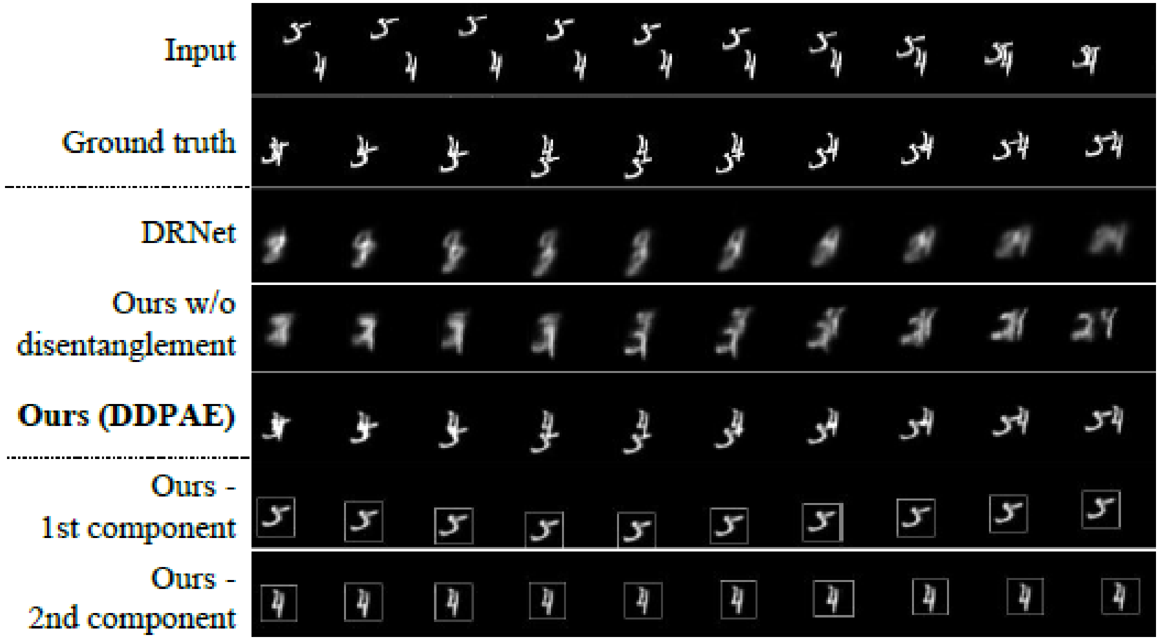



Figure 3: DDPAE separates the two digits and obtains good results even when the digits overlap. The bounding boxes of the two components are drawn manually.

Table 1: Results on Moving MNIST (Bold for the best and underline for the second best). Our results significantly outperforms the baselines.

| Model | BCE | MSE |
|---|---|---|
| Shi et al. [45] | 367.2 | - |
| Srivastava et al. [33] | 341.2 | - |
| Brabandere et al. [5] | 285.2 | - |
| Patraucean et al. [26] | 262.6 | - |
| Ghosh et al. [10] | 241.8 | 167.9 |
| Kalchbrenner et al. [15] | **87.6** | - |
| MCNet [39] | 1308.2 | 173.2 |
| DRNet [6] | 862.7 | 163.9 |
| Ours w/o Decomposition | 325.5 | 77.6 |
| Ours w/o Disentanglement | 296.1 | <u>65.6</u> |
| Ours (DDPAE) | <u>223.0</u> | **38.9** |

# 3 Experiments

## Evaluating Interdependent Components

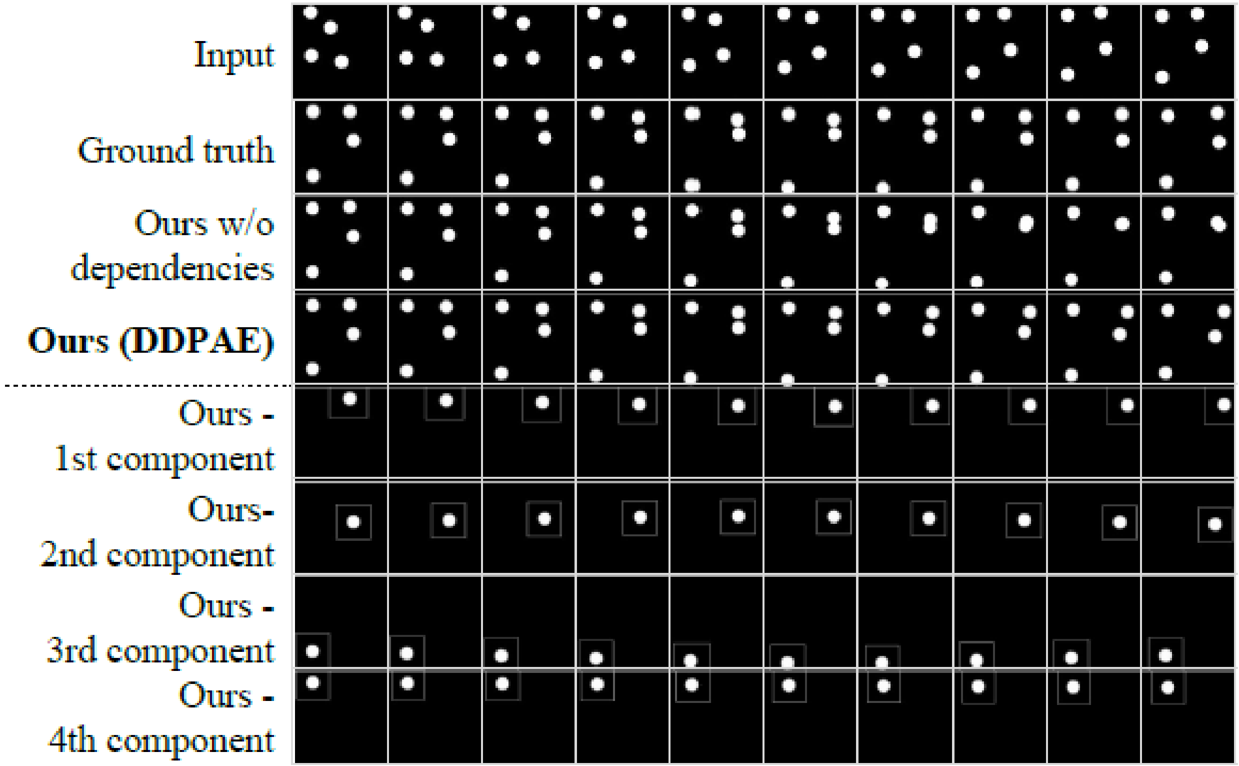



Figure 4: Our model prediction on Bouncing Balls. Note that our model correctly predicts the collision between the two balls in the upper right corner, whereas the baseline model does not.

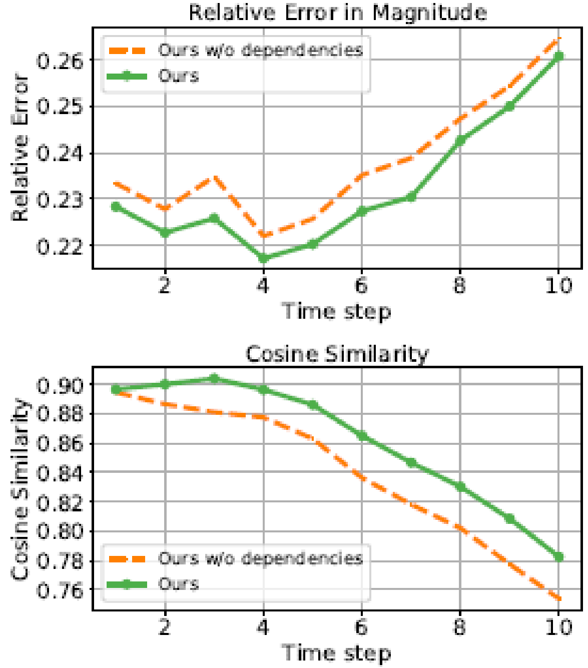



Figure 5: Accuracy of velocity with time. *Top*: Relative error in magnitude. *Bottom*: Cosine similarity.

3

# Experiments

## Evaluating Generalization to Unknown Number of Components

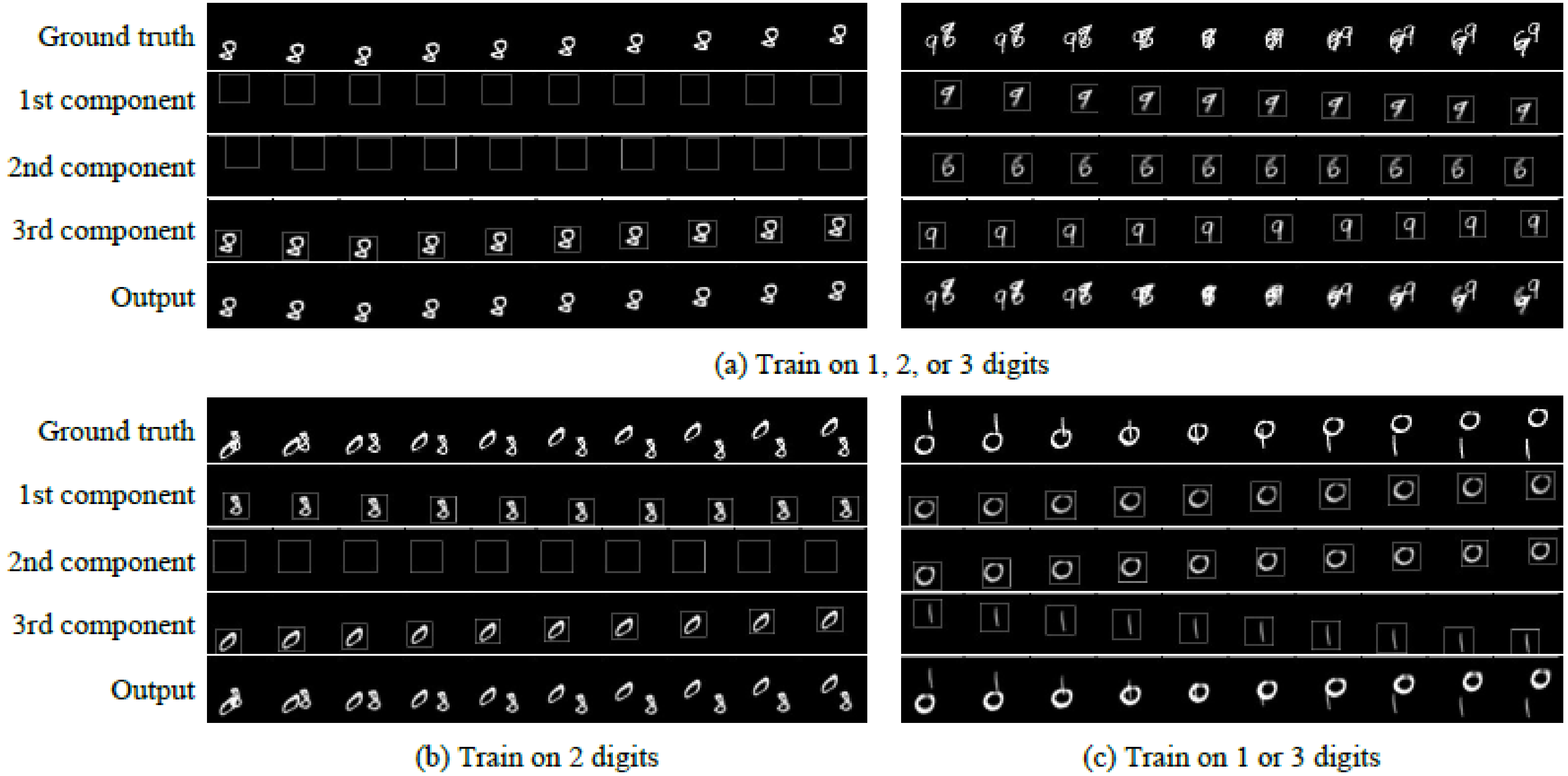



Figure 6: Results of DDPAE trained on variable number of digits. Only the predicted frames are shown. Our model is able to correctly handle redundant components.