Modelling Facial Geometry using Compositional VAEs

Timur Bagautdinov, Chenglei Wu, Jason Saragih,
Pascal Fua, Yaser Sheikh

Aman Raj
CSE-291 D
Motivation

• Photo-realistic image manipulations

• Face recognition and analysis

• Tracking of detailed 3D face models

• Image and Video editing based on reconstructed face model
Motivation for this work

• Build robust and expressive face models

• Learn multi-scale facial geometry directly from data

• Demonstrate use of deep generative models to learn meaningful geometrical representation.
Background
Supervised vs unsupervised learning

**Supervised Learning**

**Data:** $(x, y)$  
$x$ is data, $y$ is label  

**Goal:** Learn function to map  
$x \rightarrow y$  

**Examples:** Classification, regression, object detection, semantic segmentation, etc.

**Unsupervised Learning**

**Data:** $x$  
$x$ is data, no labels!  

**Goal:** Learn some hidden or underlying structure of the data

**Examples:** Clustering, feature or dimensionality reduction, etc.
Generative modeling

**Goal:** Take as input training samples from some distribution and learn a model that represents that distribution.

Density Estimation

Sample Generation

- Input samples
- Training data $\sim P_{\text{data}}(x)$
- Generated $\sim P_{\text{model}}(x)$

How can we learn $P_{\text{model}}(x)$ similar to $P_{\text{data}}(x)$?
Why generative models? Debiasing

Capable of uncovering **underlying latent variables** in a dataset

Homogeneous skin color, pose

VS

Diverse skin color, pose, illumination

How can we use latent distributions to create fair and representative datasets?
Latent variable models

Autoencoders and Variational Autoencoders (VAEs)

Generative Adversarial Networks (GANs)
Autoencoders: background

Unsupervised approach for learning a lower-dimensional feature representation from unlabeled training data.

“Encoder” learns mapping from the data, $x$, to a low-dimensional latent space, $z$.

Why do we care about a low-dimensional $z$?

MIT 6.S191
Autoencoders: background

How can we learn this latent space?
Train the model to use these features to **reconstruct the original data**

“Decoder” learns mapping back from latent, \( z \), to a reconstructed observation, \( \hat{x} \)
Autoencoders: background

How can we learn this latent space?
Train the model to use these features to **reconstruct the original data**

\[ \mathcal{L}(x, \hat{x}) = \|x - \hat{x}\|^2 \]

Loss function doesn't use any labels!!
Autoencoders for representation learning

*Bottleneck hidden layer* forces network to learn a compressed latent representation

*Reconstruction loss* forces the latent representation to capture (or encode) as much “information” about the data as possible

**Autoencoding** = *Automatically encoding* data
VAEs: key difference with traditional autoencoder

Variational autoencoders are a probabilistic twist on autoencoders!
Sample from the mean and standard dev. to compute latent sample
VAE optimization

Encoder computes: $p_\theta(z|x)$

Decoder computes: $q_\theta(x|z)$

$\mathcal{L}(\phi, \theta, x) = \text{(reconstruction loss)} + \text{(regularization term)}$
VAE optimization

Inferred latent distribution \( D \left( p_{\phi}(z|x) \perp p(z) \right) \)

Fixed prior on latent distribution

Encoder computes: \( p_{\phi}(z|x) \)

Decoder computes: \( q_{\theta}(x|z) \)

\[\mathcal{L}(\phi, \theta, x) = \text{(reconstruction loss)} + \text{(regularization term)}\]
Priors on the latent distribution

\[ D \left( p_\phi(z|x) \ Parallel p(z) \right) \]

Inferred latent distribution \hspace{1cm} Fixed prior on latent distribution

Common choice of prior:

\[ p(z) = \mathcal{N}(\mu = 0, \sigma^2 = 1) \]

- Encourages encodings to distribute encodings evenly around the center of the latent space
- Penalize the network when it tries to “cheat” by clustering points in specific regions (i.e., memorizing the data)
VAEs computation graph

Problem: We cannot backpropagate gradients through sampling layers!

Encoder computes: $p_\phi(z|x)$

Decoder computes: $q_\theta(x|z)$

$L(\phi, \theta, x) = \text{(reconstruction loss)} + \text{(regularization term)}$
Reparametrizing the sampling layer

Key Idea:

\[ z \sim \mathcal{N}(\mu, \sigma^2) \]

Consider the sampled latent vector as a sum of
- a fixed \( \mu \) vector,
- and fixed \( \sigma \) vector, scaled by random constants drawn from the prior distribution.

\[ z = \mu + \sigma \odot \varepsilon \]

where \( \varepsilon \sim \mathcal{N}(0,1) \)
Reparametrizing the sampling layer

Original form:
 bez \sim p_\phi(z|x)

Reparametrized form:
 z = g(\phi, x, \epsilon)

Deterministic node
Stochastic node
VAE summary

• Compress representation of world to something that we can use to learn

• Reparameterization trick to train end to end

• Reconstruction allows unsupervised learning (no labels!)

• Generating new examples
Generative Adversarial Networks (GANs)

Generative Adversarial Networks (GANs) are a way to make a generative model by having two neural networks compete with each other:

The generator turns noise into an imitation of the data to try to trick the discriminator.

The discriminator tries to identify real data from fakes created by the generator.
Prior Work
Global Model

• Represents all possible faces as linear combination of basis vector, e.g. by performing PCA

• $h$ is set of linear coefficients.

• $W_e$ and $W_d$ are encoding and decoding matrices

$$h = W_e \cdot M, \quad \hat{M} = W_d \cdot h$$
Local Model

• More flexible than global models as they decouple the parameters between different parts of mesh.

• \( h \) is factored into independent sets of parameters \( h_p \) for distinct patches \( M_p \) of mesh.

• \( \theta_e \) and \( \theta_d \) are parameters of encoder and decoder respectively.
Prior Work

Global Models: Designed to model whole face at once but difficult to use them to represent small details

Cons: Difficult to use them to represent small details.

Lu et al.  
Lewis et al.  
Vlasic et al.
Prior Work

Local Models: Designed to represent small details locally and more flexible

Cons: Fails to realistically represent human face at global level

Neumann et al.  Joshi et al.
Prior Work

Hybrid Models: A combination of local and global models

Cons: Fails to generalize, needs to be tailored to each individual.

Wu et al.
Prior Work (Deep Learning Approaches)

CNN that maps images into the space of facial geometry
Cons: Rely heavily on pre-defined geometry space.
Mesh Representation

• Face geometry represented as triangular mesh as pair of $(V, T)$.

• $V$ is collection of 3D vertices which encodes Geometric(Locational) information

• $T$ is set of triangles that defines the topological(Connectivity) information.
Mesh Representation

- UV parametrization of a mesh.
- Project 3D mesh into 2D image plane

\[ M \in \mathcal{R}^{H \times W \times 3} \]

- Makes it natural to perform 2D convolutions.

UV parameterization of a face. From left-to-right: x, y, z coordinates
Convolutional Mesh VAE

Parameterize

• Distribution over latent variables

$$q(z|M; \theta_e)$$

by deep net encoder $E(\cdot)$

• Generative model

$$p(M|z; \theta_d)$$

by deep net decoder $D(\cdot)$
Convolutional Mesh VAE

A variational Formulation

\[ \nu = E(M; \theta_e), \quad z \sim q(z|\nu), \quad \hat{M} = D(z; \theta_d) \]

\( \nu \) are the parameters of approximate posterior

\( z \) is latent variable that we discussed earlier, here responsible for representing shape
Compositional Mesh VAE (Why?)

• Convolutional VAE depends on low-dimensional vector $z$.

• Can't capture high frequency deformations in shape.

• Introduce multiple layers of hidden variables $z^{1:L}$

• Each layer captures separate level of detail.

$$z^{1:L} = (z_1, z_2, \ldots, z_L)$$
Compositional Mesh VAE

Joint distribution of observed meshes $M$ and latent variables $z^{1:L}$

$$p(M, z^{1:L}) = p(M|\hat{M}(z^{1:L})) \cdot \prod_{l=1}^{L} p(z^l|\xi^l)$$

$\xi^l$ are the parameters of the prior.

Posterior $q$ can be factorized as

$$q(z^{1:L}|M; \theta_e) = \prod_{l=1}^{L} q(z^l|\nu^l)$$
Compositional Mesh VAE

Factorized structure of latent space:

\[ \mathbf{\nu}^l \in \mathcal{R}^{H^l \times W^l \times C^l} \]

parameter of posterior \( q(\mathbf{z}^l | \mathbf{\nu}^l) \)

\[ \mathbf{\xi}^l \in \mathcal{R}^{H^l \times W^l \times C^l} \]

parameter of prior \( p(\mathbf{z}^l | \mathbf{\xi}^l) \)

\[ \mathbf{h}_e^l, \mathbf{\nu}^l = E^l(\mathbf{h}_{e}^{l-1}; \mathbf{\theta}_e^l) \]

\[ \mathbf{z}^l \sim q(\mathbf{z}^l | \mathbf{\nu}^l) \]

\[ \mathbf{h}_d^l, \mathbf{\xi}^l = D^l(\mathbf{h}_{d}^{l+1}, \mathbf{z}^{l+1}; \mathbf{\theta}_d^l) \]
Model Fitting

• Given image data \( X \), goal is to find parameter vectors \( z^{1:L} \) such that decode mesh \( \hat{M} \) fits data as well as possible.

• Equivalent to solving following MAP(Maximum a posterior Probability)

\[
\log p(X|\hat{M}(z^{1:L})) + \sum_{l=1}^{L} \log p(z^l|\xi^l(z^{l+1}))
\]

• Different \( p(X|\hat{M}) \) for different types of input data
Modeling: 3D to 3D correspondences

we know the position \( M_i \) of a subset \( \mathcal{I} \) of vertices

\[
\sum_{i \in \mathcal{I}} \log p(M_i | \hat{M}_i) \propto - \sum_{i \in \mathcal{I}} \| M_i - \hat{M}_i \|_2^2
\]
Modeling: 2D to 3D correspondences

let $\mathcal{I}$ now be the set of vertices $M_i$

for which we have 2D projections $P_i \in \mathcal{R}^2$

\[
\sum_{i \in \mathcal{I}} \log p(P_i | \hat{M}_i) \propto -\sum_{i \in \mathcal{I}} \|P_i - \Pi_{K,R|t}\hat{M}_i\|_2^2
\]

camera intrinsic $K \in \mathcal{R}^{3\times3}$ and extrinsic $R|t \in \mathcal{R}^{3\times4}$ parameters

$\Pi_{K,R|t}\hat{M}_i$ are the 2D projections of the model
Modeling: Depth maps

Set of vertices visible from depth camera camera point of view

\[ \mathcal{I}_V \subset H \times W \]

Depth Maps projected from 3D vertex position using camera extrinsic

\[ \hat{D}_i = (R \cdot \hat{M}_i + t) \]

\[ \sum_{i \in \mathcal{I}_V} \log p(D_i | \hat{M}_i) \propto - \sum_{i \in \mathcal{I}_V} \left\| D_i - \hat{D}_i \right\|^2_2 \]
Modeling: Shape from shading

• Convert meshes into RGB images

• Assume simple Lambertian model with single 3-channel light source $L$. 

• Compute vertex normals $\hat{N}$ from mesh $\hat{M}$, given texture $T$.

\[
\hat{I}_i = T_i \cdot L \cdot \hat{N}_i.
\]

\[
\sum_{i \in I_V} \log p(I_i | \hat{M}_i) \propto - \sum_{i \in I_V} ||I_i - \hat{I}_i||^2_2
\]
Evaluation

Dataset:

• Stereo based 3D face reconstruction

• Captured 20 different people, each performing a set of expression

• Total 2140 high-quality meshes, split as 1712 for training and 428 for testing.
Evaluation

• Baseline:
  LINEAR: Traditional linear model
  VAE: Convolutional VAE

• Implementation:
  Adam with learning rate = 0.0001

• Evaluate on 3D vertices for different proportion of correspondences trained on.

• Results range from using only $z^1$ (less priors) which influence overall shape to $z^{1:4}$ (more priors) influence fine details.
Evaluation: 3D to 3D correspondences

0.2% amount to 60 3D to 3D correspondences.

<table>
<thead>
<tr>
<th>Method</th>
<th>0.2%</th>
<th>0.5%</th>
<th>2%</th>
<th>10%</th>
</tr>
</thead>
<tbody>
<tr>
<td>LINEAR</td>
<td>2.795</td>
<td>1.309</td>
<td>1.016</td>
<td>0.980</td>
</tr>
<tr>
<td>VAE</td>
<td>1.678</td>
<td>1.317</td>
<td>1.176</td>
<td>1.139</td>
</tr>
<tr>
<td>OURS $z^1$</td>
<td>1.470</td>
<td>1.079</td>
<td><strong>0.596</strong></td>
<td><strong>0.247</strong></td>
</tr>
<tr>
<td>OURS $z^{1:2}$</td>
<td>1.468</td>
<td>1.121</td>
<td>0.609</td>
<td>0.336</td>
</tr>
<tr>
<td>OURS $z^{1:3}$</td>
<td>1.396</td>
<td>1.020</td>
<td>0.616</td>
<td>0.467</td>
</tr>
<tr>
<td>OURS $z^{1:4}$</td>
<td><strong>1.320</strong></td>
<td><strong>0.986</strong></td>
<td>0.775</td>
<td>0.717</td>
</tr>
</tbody>
</table>

RMSE in mm for different proportions of constrained vertices.
Evaluation: 2D to 3D correspondence

Errors are higher than 3D-3D due to ambiguities in 2D-3D correspondences

<table>
<thead>
<tr>
<th>Method</th>
<th>0.2%</th>
<th>0.5%</th>
<th>2%</th>
<th>10%</th>
</tr>
</thead>
<tbody>
<tr>
<td>LINEAR</td>
<td>4.381</td>
<td>3.691</td>
<td>3.394</td>
<td>3.302</td>
</tr>
<tr>
<td>VAE</td>
<td>3.606</td>
<td>3.183</td>
<td>3.114</td>
<td>3.077</td>
</tr>
<tr>
<td>OURS $z^1$</td>
<td>2.690</td>
<td>2.521</td>
<td><strong>2.390</strong></td>
<td><strong>2.330</strong></td>
</tr>
<tr>
<td>OURS $z^{1:2}$</td>
<td>2.660</td>
<td>2.521</td>
<td>2.396</td>
<td>2.343</td>
</tr>
<tr>
<td>OURS $z^{1:3}$</td>
<td>2.606</td>
<td><strong>2.512</strong></td>
<td>2.431</td>
<td>2.396</td>
</tr>
<tr>
<td>OURS $z^{1:4}$</td>
<td><strong>2.586</strong></td>
<td>2.545</td>
<td>2.472</td>
<td>2.453</td>
</tr>
</tbody>
</table>

RMSE in mm for different proportions of constrained vertices
Evaluation: Depth Maps

- Corrupt the depth maps with different levels of Gaussian noise to model real life situation

<table>
<thead>
<tr>
<th>Method</th>
<th>$\sigma^2 = 1$</th>
<th>$\sigma^2 = 2$</th>
<th>$\sigma^2 = 3$</th>
</tr>
</thead>
<tbody>
<tr>
<td>LINEAR</td>
<td>3.908</td>
<td>3.924</td>
<td>3.953</td>
</tr>
<tr>
<td>VAE</td>
<td>3.167</td>
<td>3.199</td>
<td>3.249</td>
</tr>
<tr>
<td>OURS $z^1$</td>
<td>3.032</td>
<td>3.142</td>
<td>3.252</td>
</tr>
<tr>
<td>OURS $z^{1:2}$</td>
<td><strong>3.020</strong></td>
<td><strong>3.114</strong></td>
<td>3.215</td>
</tr>
<tr>
<td>OURS $z^{1:3}$</td>
<td>3.079</td>
<td>3.127</td>
<td><strong>3.191</strong></td>
</tr>
<tr>
<td>OURS $z^{1:4}$</td>
<td>3.110</td>
<td>3.150</td>
<td>3.226</td>
</tr>
</tbody>
</table>

RMSE in mm for different noise levels
Results

Visual results for fitting noisy depth

From left to right: input depth map, rendered mesh (Linear), rendered mesh (Proposed), rendered mesh (Proposed) overlaid with the image.
Results

Visual results for shape from shading.

From left to right: rendered mesh (Linear), rendered mesh (Proposed)
Results

Visualizing the effect of varying first PCA component of $z^{1:2}$ (top) and $z^{4:5}$ (bottom)
Results

The leftmost and rightmost columns are original meshes.
Top: interpolating $z^{1:2}$, keeping $z^{5:6}$ fixed. Bottom: interpolating $z^{5:6}$, keeping $z^{1:2}$ fixed.
Results
Critique of Experiment and Follow up

• No comparison with previous state of the art on any task.

• Fails to provide information about how do they learn the probabilistic connection in their compositional VAE.

• Investigating alternative architectures in decoders such as PixelRNN: predicting image pixel by pixel
Thank You!
Questions?