Multi-view Convolutional Neural Networks for 3D Shape Recognition

by

Hang Su Subhransu | Maji Evangelos Kalogerakis | Erik Learned-Miller
University of Massachusetts, Amherst

Presented by

Johannes Andersen
Overview

• Problem
• Solution
• Usecases
• Conclusion
• TLDR
Problem

• Representing and recognizing 3D models in CNN’s:
  • High dimensionality (overfitting)
  • Lower resolution (compared to 2D)
  • Fewer viable image descriptors available in 3D
  • Fewer and smaller pretrained datasets than in 2D
  • Even rendering 3D shape at 1 view was better than 3D recognition
Solution

3D shape model rendered with different virtual cameras

2D rendered images

our multi-view CNN architecture

output class predictions
Solution

• Step one: Acquire 2D images of model from several views
• Step two: Create descriptor by CNN based on views (MVCNN)
• Step three:
  • Recognize 3D shapes using another CNN
  • Recognize sketches
  • 3D shape retrieval
Use case: Recognizing 3D shapes

- Single-view: 78.8% acc.
- 12-view: 89.5%
- 3D descriptor: 77.3% acc.
Usecase: Sketch recognition ("jittering")

- VGG-M: 77.3% acc.
- VGG-VD (deeper): 86.0% acc.
- MVCNN VD: 87.2% acc.
Usecase: 3D shape retrieval

- Red boxes are wrong
Conclusion

- 2D input to modern architectures outperform 3D representations
- Better accuracy through aggregated descriptors
- Can use these descriptors to find 3D models of 2D images

- Which views are most informative?
- How many views are needed?
- Can it be used in real-time with video?
- Can it select useful views on the fly?
TLDR

1. **3D model**: Render to multiple views
2. **2D images**: CNN1 generates descriptor
3. **Descriptors**: CNN2 classifies the descriptor
4. **Class**:
Sources


• Original implementation: https://github.com/suhangpro/mvcnn

• TensorFlow implementation: https://github.com/WeiTang114/MVCNN-TensorFlow