AI3064 Computer Vision

Final Project

Proposal Due: Oct. 12 2023

Report Due: Jan. 1, 2023

The objective of the final project is to make use of what you have learned during this course to explore potential solutions to problems using computer vision. The final deliverables will be

- A written report, which is limited to eight pages, including figures and tables, in the CVPR style. Additional pages containing only cited references are allowed.
- Source code of your implementation, which should be executable.

You will work in **groups of 4 or 5**. To facilitate forming teams, we will create a team search form on Canvas. You could either look for a team or look for more students.

Only one member of your team will need to submit your work on Canvas, but make sure that this member successfully adds all team members on Canvas.

Below we will define some example projects that you can work on, but feel free to propose other projects that you find interesting. The goal of the project is to explore and to compare different approaches to whatever problem you choose to work on. Grading will reflect the thoroughness and quality of that exploration.

We will ask the following questions:

- How challenging is this problem?
- What approaches did you take to solve this problem, and why?
- How did you explore the space of solutions -- architectures, hyperparameters, training methods etc. -- your chosen approach(es) presented?
- How did you evaluate the performance of the final approach(es) you investigated?

The key point here is that we're not just interested in the absolute performance of what you produce, but also the thought process that led you there. Ideally, you will have done some comparative performance analysis to back up the choices you made.

Grading will be based on the completeness of the project, the clarity of the writeup, the level of complexity/difficulty of the project, and your ability to justify the choices you made.

Schedule

Oct. 12: By this date, we should know who is working together, and we should have a short (no more than 1 page) project proposal.

Last two weeks of class: Each team makes a short (< 10 minute) presentation on your project.

Jan. 1 (TBD): Final deliverable submission due.

Final Report Format

Your final report should document the following:

Introduction

Related work, Methodology, Experimental results, and Conclusion.

Project Proposal Template

Your 1-page-long project proposal should include the following:

- Team member list
- Project description: 2-3 paragraphs including:
    - A definition of your problem.
    - Some details of how you propose to solve your chosen problem including a breakdown into approaches/components.
    - The datasets you’ve identified (to the extent needed for your project).
    - At least one (ideally 2-3) paper or similar expositions that form a starting point for your investigation.
- Team member assignments (who does what).

We are happy to entertain some “special” projects that do not fit this template if you already have something specific in mind where you’ve done some initial work, but in that case you need to give us a written proposal which outlines the scope of the project and your goals.

Example Projects

Anomaly Segmentation

Anomaly segmentation is a crucial task for safety-critical applications, such as autonomous driving in urban scenes, where the goal is to detect out-of-distribution (OOD) objects with categories which are unseen during training. For example, there are 19 categories in the standard training set of CityScapes. When the models trained on the training set are encountered with unseen objects, they will classify them into some categories they have learned in the training process. For example, they may recognize an obstacle as a part of a road, which is intolerable in autonomous driving. So it is critical to finding out a way to solve such a problem.

In this project, your task can be concluded as follows:

What you already have: a trained semantic segmentation model like DeepLabv3+ and its corresponding training dataset like CityScapes. (You may train such model by yourself or download it from github.)

What you need to do: design an anomaly segmentation model, which can segment out the OOD (anomalous) objects of the above-mentioned model.

Your evaluation metric: AP, AUROC, FPR95 (You can find them in the reference)

Your datasets: FishyScapes: you can find its description on the official website (https://fishyscapes.com)

Tips:

You can choose one or more datasets to evaluate your model. It is encouraged to adopt more datasets. Some other datasets are accepted.

Both training a model without modifying the original segmentation model and training a segmentation model which can segment out anomaly are accepted. However, we believe the former is better.

In your final report, you should submit executable codes. The results in your report must be reproducible.

References:

- Dan Hendrycks and Kevin Gimpel. A baseline for detecting misclassified and out-of-distribution examples in neural networks. Proceedings of International Conference on Learning Representations, 2017.
- Sanghun Jung, Jungsoo Lee, Daehoon Gwak, Sungha Choi, and Jaegul Choo. Standardized max logits: A simple yet effective approach for identifying unexpected road obstacles in urban-scene segmentation. In Proceedings of the IEEE/CVF International Conference on Computer Vision, pages 15425–15434,2021.

- Robin Chan, Matthias Rottmann, and Hanno Gottschalk. Entropy maximization and meta classification for out-of-distribution detection in semantic segmentation. In Proceedings of IEEE international conference on computer vision (ICCV), 2021.
- David Haldimann, Hermann Blum, Roland Siegwart, and Cesar Cadena. This is not what i imagined: Error detection for semantic segmentation through visual dissimilarity. arXiv preprint arXiv:1909.00676, 2019.
- Krzysztof Lis, Krishna Nakka, Pascal Fua, and Mathieu Salzmann. Detecting the unexpected via image resynthesis. In Proceedings of IEEE international conference on computer vision (ICCV), 2019.
- Xia Yingda, Zhang Yi, Liu Fengze, Shen Wei, and Yuille Alan. Synthesize then compare: Detecting failures and anomalies for semantic segmentation. In Proceedings of the European Conference on Computer Vision (ECCV), 2020.
- Giancarlo Di Biase, Hermann Blum, Roland Siegwart, and Cesar Cadena. Pixel-wise anomaly detection in complex driving scenes. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 16918– 16927, 2021.

Indoor scene reconstruction

Scene reconstruction is the process of reconstructing a digital version of a real-world object from pictures or scans of the object. It is a very complex problem with many research histories, open problems, and possible solutions. Currently, most approaches to this problem can be broken down into two broad phases: first, you take your input and turn it into a cloud of points on the object's surface, then reconstruct a mesh from this point cloud. These approaches are summarized as SFM (structure from motion) and MVS (multi-view stereo). Recently, neural rendering-based methods are also popular in reconstructing scenes. Neural rendering is an emerging class of deep image, and video generation approaches that enable control of scene properties such as illumination, camera parameters, pose, geometry, appearance, and semantic structure. It combines machine learning techniques with physical knowledge from computer graphics to obtain controllable and photo-realistic models of scenes.

In this project, your task can be concluded as follows:

What you already have: calibrated cameras' positions and rotations, posed scene images (RGB or RGB-D), any on-the-shelf pretrained model.

What you need to do: reconstruct indoor scenes with respect to mesh.

Your evaluation metric: accuracy, completeness, precision, recall and F-score. (You may find the definition of these metrics in Atlas)

Your datasets: Replica. Replica is a reconstruction-based 3D dataset of 18 high fidelity scenes with dense geometry, HDR textures, and semantic annotations. A separate metrics report is encouraged. You can download the entire data from Omnidata (https://github.com/EPFL-VILAB/omnidata/tree/main/omnidata_tools/dataset#readme).

Tips:

You can choose one or more datasets to evaluate your model. It is encouraged to adopt more datasets. Some other datasets are accepted.

Both RGB-based methods and RGB-D-based methods are acceptable. However, RGB-D-based methods would be more demanding.

Images are provided with poses, while there would be a bonus if you estimate the poses from images and compare them with the ground truth.

In your final report, you should submit executable codes. The results in your report must be reproducible. A detailed table for performance among different scenes must be in the report. Reconstructed meshes ought to be attached as supplementary for visualization.

References:

- Murez, Zak, et al. "Atlas: End-to-end 3d scene reconstruction from posed images." European conference on computer vision. Springer, Cham, 2020.
- Schonberger, Johannes L., and Jan-Michael Frahm. "Structure-from-motion revisited." Proceedings of the IEEE conference on computer vision and pattern recognition. 2016.
- Yao, Yao, et al. "Mvsnet: Depth inference for unstructured multi-view stereo." Proceedings of the European conference on computer vision (ECCV). 2018.
- Huang, Po-Han, et al. "Deepmvs: Learning multi-view stereopsis." Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2018.
- Sucar, Edgar, et al. "iMAP: Implicit mapping and positioning in real-time." Proceedings of the IEEE/CVF International Conference on Computer Vision. 2021.
- Zhu, Zihan, et al. "Nice-slam: Neural implicit scalable encoding for slam." Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2022.
- Azinović, Dejan, et al. "Neural RGB-D surface reconstruction." Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2022.
- Yariv, Lior, et al. "Volume rendering of neural implicit surfaces." Advances in Neural Information Processing Systems 34 (2021): 4805-4815.
- Wang, Peng, et al. "Neus: Learning neural implicit surfaces by volume rendering for multi-view reconstruction." arXiv preprint arXiv:2106.10689 (2021).
- Eftekhar, Ainaz, et al. "Omnidata: A scalable pipeline for making multi-task mid-level vision datasets from 3d scans." Proceedings of the IEEE/CVF International Conference on Computer Vision. 2021.

Handwritten mathematical expression recognition

Handwritten mathematical expression recognition (HMER) aims to generate the corresponding LATEX sequence from a handwritten mathematical expression image. The recognition of handwritten mathematical expressions has led to many downstream applications, such as online education, automatic scoring, and formula image searching. Different from traditional text recognition methods, HMER not only needs to recognize all characters, but also needs to identify and understand the order and relationships between latex symbols (e.g., "^", "_", "{", and "}".) and contexts. Existing methods can be summarized into sequence-based and tree-based. Sequence-based decoder outputs character results sequentially by extracting important regions contributed to character recognition. The tree-based decoder takes a more structured approach to text recognition, representing the relationships between symbols as a tree or hierarchical structure.

In this project, your task can be concluded as follows:

What you already have: a trained basic recognizing model, such as CoMER, SAN and CAN. (You may train such models by yourself or download them from Github.)

What you need to do: design an advanced expression recognition model, which can better model the relationships between symbols and contexts, and thus recognize the handwritten mathematical expression more accurately.

Your evaluation metric: ExpRate, Errors (Edit Distance).

Your datasets: CROHME 14, CROHME 16, CROHME 19: you can find its description on the official website (CROHME 2019 Home Page (rit.edu))

Tips:

You can choose one or more datasets to evaluate your model. It is encouraged to adopt more datasets. Some other datasets are accepted.

Both Sequence-based methods and Tree-based methods are acceptable. While recognition results are important, we encourage exploring other effective recognition solutions based on the essential characteristics of the handwritten mathematical expression.

In your final report, you should submit executable codes. The results in your report must be reproducible. A detailed table for performance among different datasets must be in the report.

References:

- Wenqi Zhao and Liangcai Gao. CoMER: Modeling Coverage for Transformer-based Handwritten Mathematical Expression Recognition. In Proceedings of the European Conference on Computer Vision (ECCV), 2022.
- Bohan Li, Ye Yuan and etc. When Counting Meets HMER: Counting-Aware Network for Handwritten Mathematical Expression Recognition. In Proceedings of the European Conference on Computer Vision (ECCV), 2022.
- Ye Yuan, Xiao Liu, Wondimu Dikubab and etc. Syntax-Aware Network for Handwritten Mathematical Expression Recognition. In Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2022

- Wenqi Zhao , Liangcai Gao, Zuoyu Yan, Shuai Peng, Lin Du, and Ziyin Zhang. Handwritten Mathematical Expression Recognition with Bidirectionally Trained Transformer. In Proceedings of International Conference on Document Analysis and Recognition (ICDAR), 2021