DEEP LEARNING FOR VISUAL RECOGNITION

Practical information about the course project

Henrik Pedersen, PhD
External lecturer
Department of Computer Science
Aarhus University
hpe@cs.au.dk

Important next steps

- Form groups
  - 2 or 3 people
  - Enroll in one of the predefined groups on Brightspace (follow instructions under “Week 2”)
- Create a private Slack channel for your group
  - Use the same group number as on Brightspace!
  - Invite TA Josefine, TA Shakil, and Lecturer Henrik
- Submit your project proposal
  - There is a group assignment on Brightspace
  - **Deadline Friday, September 26th**

Two types of projects

- The goal of the course project is to demonstrate that you can apply what you have learned in class to a problem of your interest.
- Potential projects *usually* fall into these two tracks:
  - **Applications**: Pick a real-world problem and apply neural networks to solve it.
  - **Models**: Build a new model/algorithm with neural networks, or a new variant of models, and apply it to solve vision tasks.
- Get inspiration from Papers with Code, Daily AI Papers, Yannic Kilcher's YouTube channel, and Stanford cs231n

Requirements

- Your project should involve pixels of visual data in some form.
- You must train a neural network model, evaluate it and perform experiments to improve its performance.
- When designing your project, keep the deep learning workflow in mind:
  - Acquire and prepare a dataset for deep learning
  - Set up an appropriate deep learning model (i.e., a neural network) to solve the task at hand
  - Train and test your model with the correct evaluation metrics
  - Perform motivated experiments to improve your model's performance (e.g., reduce overfitting)
  - Explain your results
- You should demonstrate that you are familiar with all of the steps above.
- Focus on the deep learning part, not the application/system around it!
- Sharing code and copying code from places like GitHub is okay, but you must add some of your own work on top of other people’s work.

Hand-in 1: Project proposal

- No more than one page:
  - Project title
  - Group number
  - List of group members
  - Purpose – which problem do you wish to solve? Or which ConvNet model do you wish to modify?
  - Which data set(s) are you planning on using?
  - Thoughts on methods – how do you intend to solve your problem? Or how will you modify ConvNets?
  - Which experiments are you planning on doing?
  - References to 1-3 key research papers of related work
- **Deadline:** Friday, September 26th
- **Important:** Your proposal is not a "contract". You will encounter unforeseen challenges and get unexpected results from your experiments. As a result, you might have to change the scope of your project as you go along. That's just a natural part of the process.
- **My best advice:** Start now!

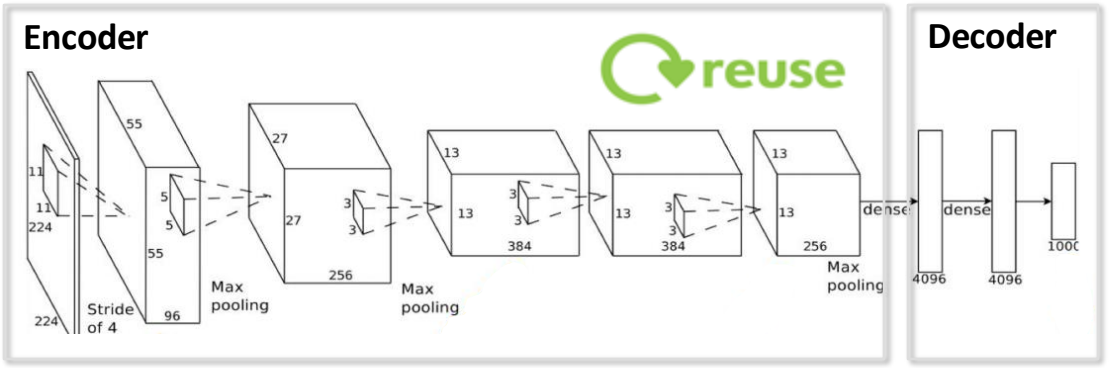
Hand-in 2: Written report

- Use this arXiv template
- Report should be structured like a research paper:
  - Abstract
  - Introduction (Includes motivation and objectives of your project. Be realistic and be specific!)
  - Related work (Summarize at least 2-3 key references. What makes your approach different?)
  - Methods (What did you do and how? Describe network architecture, data set, experiments, etc.)
  - Results (Objectively summarize your results and compare to related work or state-of-the-art)
  - Discussion (How well did you do? What worked? What didn't work?)
- Final report between 6-8 pages, including figures and references.
- Do not exceed the page limits!
- **New:** Hand in 5-minute video presentation (group-level)
- **Deadline:** December 5th

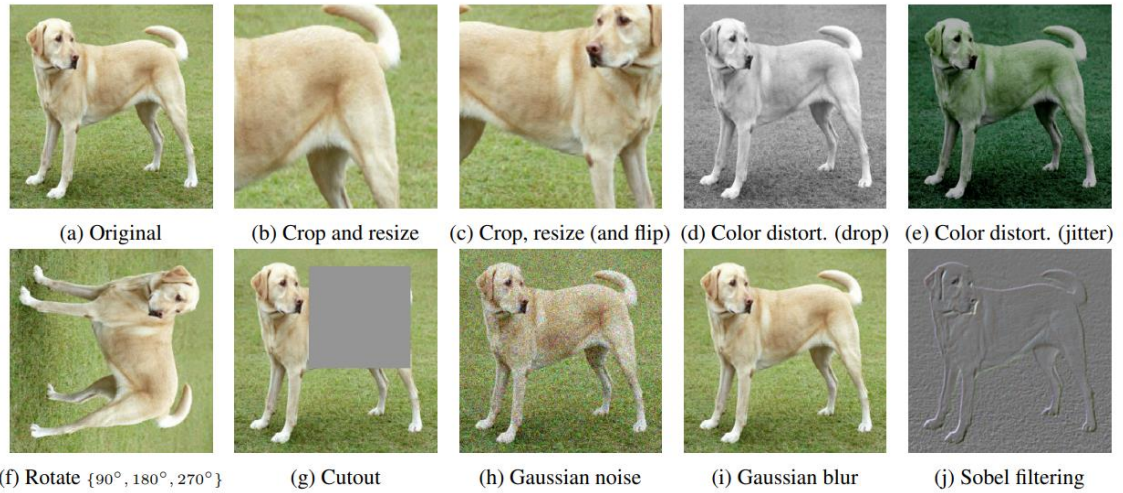
Example – custom image classifier

- Task is to recognize three types of Russian tanks
  - BMD4, BMP2, and T72
- Dataset
  - Self-made from Google Image search
  - 500 images from each class
- Neural network architecture: ResNet50
- Training:
  - Transfer learning using ResNet50 backbone (encoder) pre-trained on ImageNet
  - Data augmentation
- Evaluation / analysis
  - Inspect loss curves
  - Confusion matrix
  - Qualitative assessment of mis-classified images
  - Heatmaps (GradCAM – covered in Lecture 10)

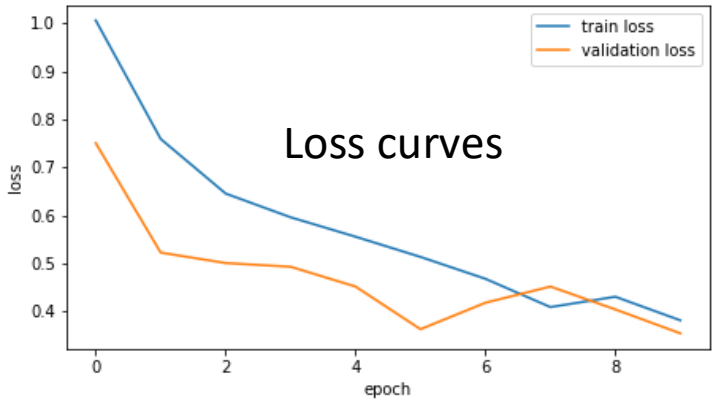
Transfer learning



Data augmentation

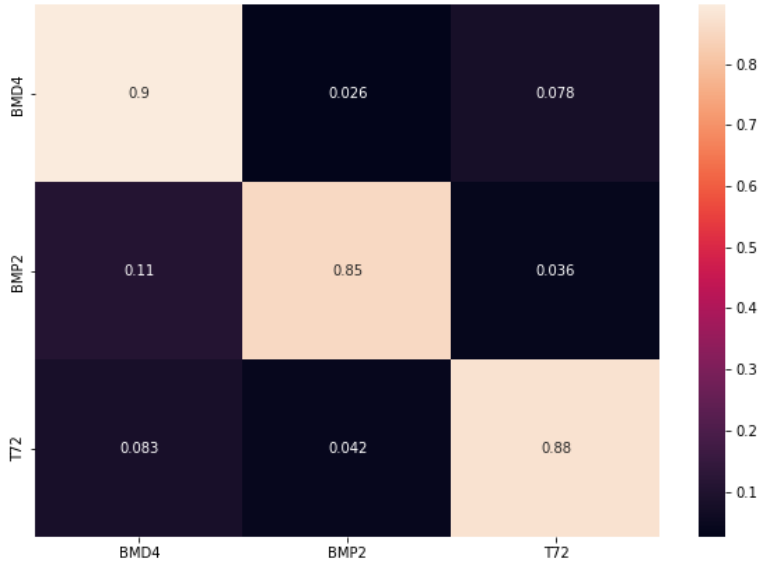


Example – custom image classifier



Mis-classified images

Confusion matrix



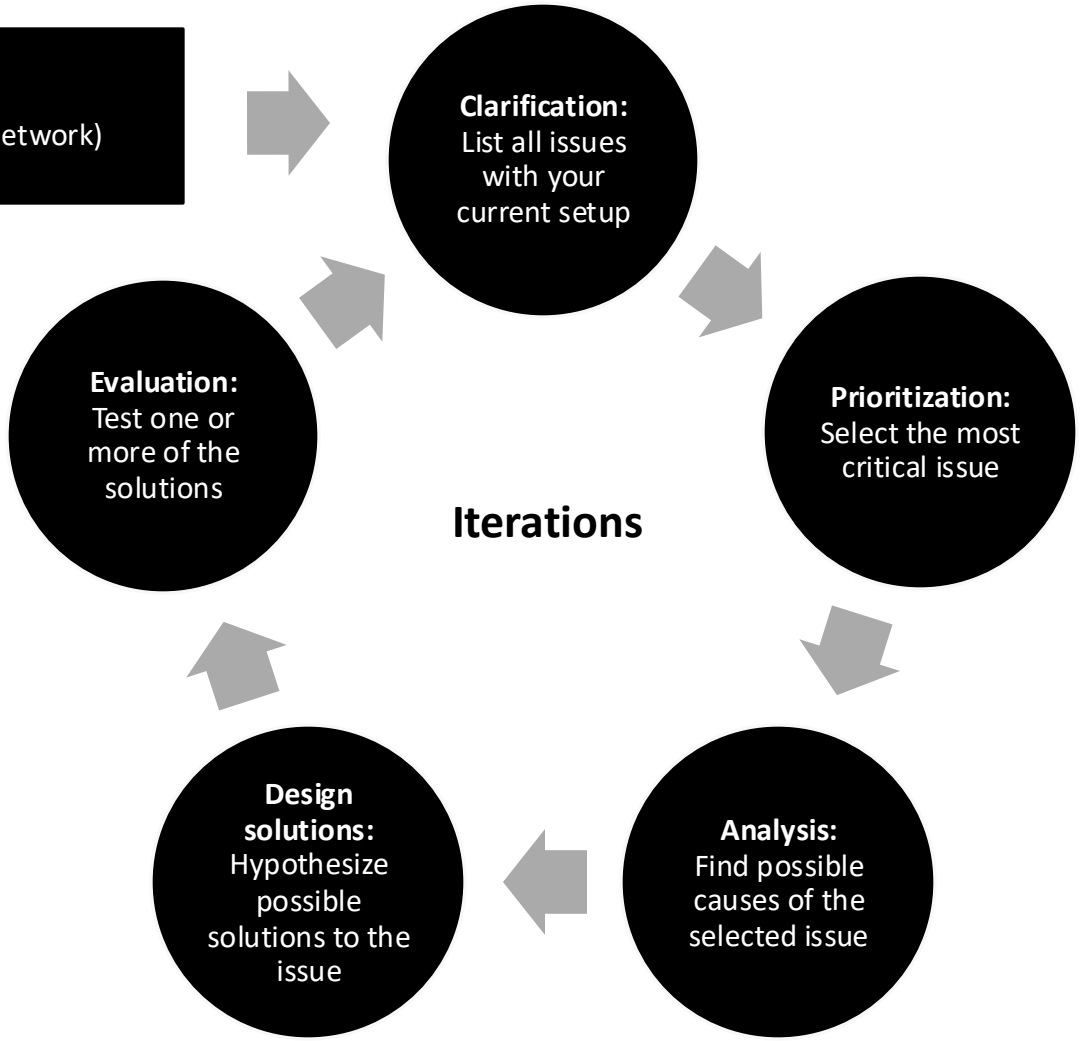
Heatmaps

General advise

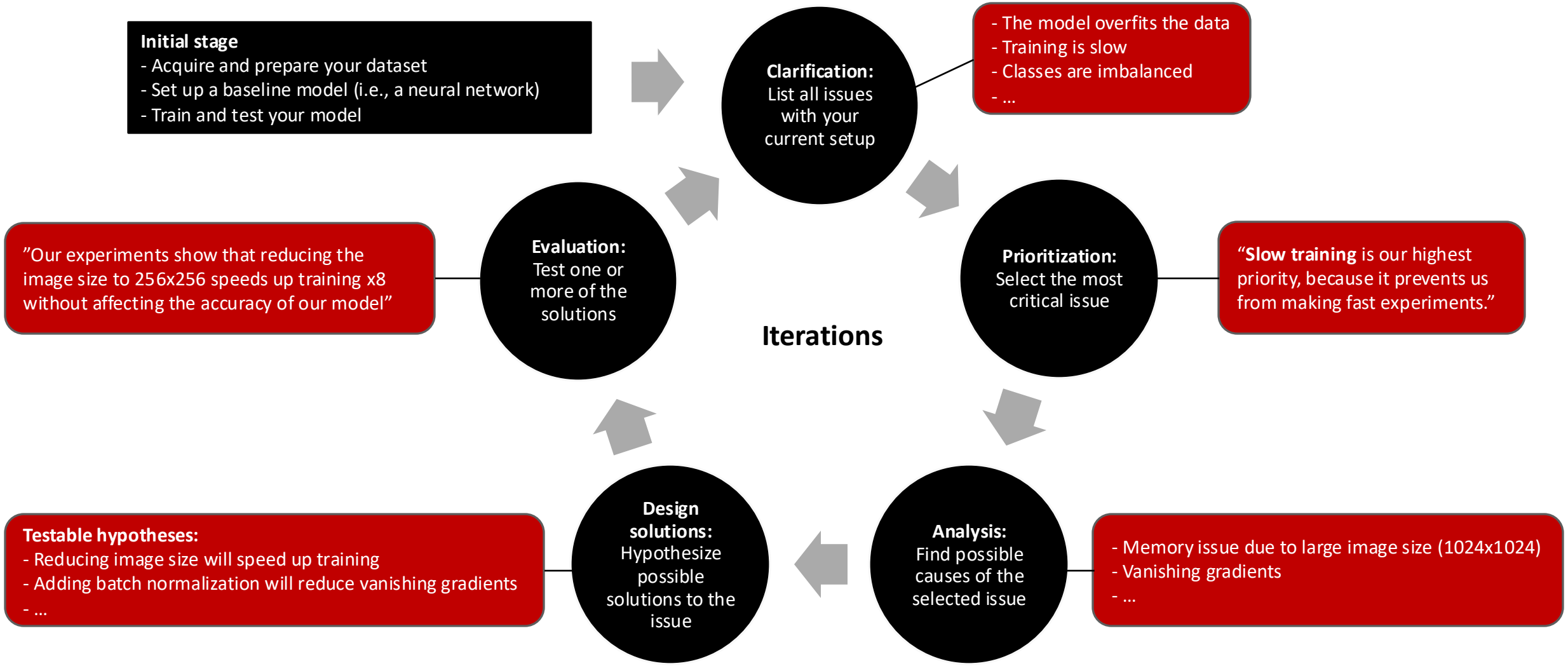
- **Initial stage:** Create a baseline model (e.g., an image classifier) and evaluate its performance.
- **Iterations:** Perform motivated adjustments to your setup (e.g., neural network architecture) to address potential issues and/or improve model performance.
- Inspecting and understanding your dataset
  - Do you have enough training images?
  - Where do the labels come from, and can you trust them?
  - Are the classes balanced (only relevant if training a classifier)?
- Spotting overfitting
  - Inspect the loss curves
  - Confusion matrix
  - Heatmaps
- Addressing overfitting and class imbalance
  - Data augmentation
  - Regularization techniques (weight decay, etc.)

Typical workflow

Initial stage
- Acquire and prepare your dataset
- Set up a baseline model (i.e., a neural network)
- Train and test your model



Typical workflow



What characterizes a **good report**?

- A **thorough Related Work section** showing evidence of a proper literature study.
- **Choices are explained and motivated**, demonstrating understanding of the theory.
- The project goes **beyond lecture material**, showing ability to **acquire and apply new knowledge** independently.
- Report includes **dataset description and visualizations**, so the reader can judge data characteristics and challenges.
- Discussion of **overfitting** and other key challenges in deep learning, ideally with measures taken to address them.
- Clear balance between “what” was done and “why” it was done.
- Avoids parroting lecture theory; instead, demonstrates ability to **apply, compare, and relate concepts** (SOLO taxonomy).
- Cites only papers that were actually read.

What characterizes a **weak report**?

- Report does not reflect expected workload (≈112 hours per student).
- Focuses mainly on **“what” was done** with little to no **justification (“why”)**.
- **Missing dataset information** (e.g., sample size, class distribution, train/validation split).
- **No dataset images**, making it unclear how challenging the problem is.
- **Important concepts omitted** (e.g., overfitting not mentioned or addressed).
- Inclusion of **irrelevant or unread citations**.
- Heavy reliance on **reproducing theory from lectures** without demonstrating application.

Download report examples via this link:
https://brightspace.au.dk/content/enforced/183737-LR50323/report%20examples.zip?ou=183737

Common pitfall

- There are many “plug-and-play” tools that make it easy to train a neural network on your own dataset with just a few clicks.
- These are great for building real-world applications, but not well suited for a course project because:
  - most of the important design choices have already been made for you,
  - the experimentation is usually reduced to trivial hyperparameter tuning,
  - and this gives you very little room to show the deeper skills that a strong report should demonstrate.
- Example:
  - YOLO Object Detector
  - If you want to work with object detection, it’s often better to implement and train your own detector rather than relying on ready-made solutions.
  - In a course project, you will typically only have time to implement a small part of the full detection pipeline – and that’s perfectly fine. The key is to focus on that one component, explore it in depth, and make it work well. If you achieve good results earlier than expected, you can always broaden the scope of your project.

Types of projects from previous years I

- **Image Classification or Image Regression (less frequent)**
  - **Description:** Train a CNN (e.g., ResNet, MobileNet) to classify images from a dataset of your choice (birds, dogs, traffic signs, plant species, food, skin lesions, X-rays, age estimation from faces, etc.).
- **Image Segmentation**
  - **Description:** Use U-Net or similar architectures to segment objects of interest (tumors or organs in medical scans, image tampering localization (i.e., deep fakes), buildings in satellite imagery, etc.).
- **Object Detection (usually a Partial Pipeline)**
  - **Description:** Implement part of an object detection pipeline (e.g., region proposal, bounding box regression, or fine-tuning YOLO).
- **Domain Adaptation**
  - **Description:** Train a model on one dataset (e.g., synthetic images) and test it on another (real images).
- **Image Generation using GANs / VAEs / Diffusion Models**
  - **Description:** Use a generative model to create new images (e.g., colorize grayscale photos, generate new samples, style transfer, converting synthetic images to real images using CycleGAN).

Types of projects from previous years II

- **Image Captioning (Vision + Language)**
  - **Description:** Combine a CNN (for image features) with an RNN/Transformer (for text generation) to generate captions for images.
- **Monocular Depth Estimation**
  - **Description:** Train a network to estimate depth from a single image.
- **Adversarial Robustness**
  - **Description:** Investigate how models fail under adversarial attacks and explore defenses like adversarial training.
- **Active Learning / Data Efficiency**
  - **Description:** Show how models can learn effectively from fewer labels by iteratively selecting the most informative samples for annotation.
- **Lightweight Models for Deployment**
  - **Description:** Train and optimize a small model (e.g., MobileNet, pruning, quantization) for edge devices.
- **Few-Shot Learning**
  - **Description:** Train a classifier with very limited labelled data using pre-trained models, prototypical networks, or fine-tuning.

Inspiration

SELF-STUDY

1. Medical Image Segmentation

- Medical image segmentation is the task of segmenting objects of interest in a medical image - for example organs or lesions.
- https://paperswithcode.com/task/medical-image-segmentation
- Possible project
  - Find a public data set for medical image segmentation.
  - Apply a standard U-Net or possibly Attention U-Net and see how well it works.
  - See if you can increase performance by
    - Using transfer learning
    - Modifying the network architecture
    - Modifying the loss function
    - …
- Also possible to find other applications than medical data, including visual inspection of steel and satellite imagery.

2. Domain adaptation

- Domain adaptation (DA) is the task of training a model in one domain (source domain) and applying it in a different domain (target domain).
- **Motivation:** Suppose you have very limited data from the target domain, but you have tons of data from a similar domain (the source domain). Can we utilize the data from the source domain to make a model that works well in the target domain?
- **Example:** Train an object classifier on synthetic images generated using computer graphics, and make it work on real images (not as easy as it sounds). See examples here:
  - http://openaccess.thecvf.com/content_ECCVW_2018/papers/11129/Hinterstoisser_On_Pre-Trained_Image_Features_and_Synthetic_Images_for_Deep_Learning_ECCVW_2018_paper.pdf
  - https://towardsdatascience.com/deep-learning-with-synthetic-data-will-make-ai-accessible-to-the-masses-15b99343dd0e
- https://paperswithcode.com/task/domain-adaptation
- Possible project
  - Find a public data set (like this) of both real and synthetic images of objects.
  - Train and test a DA model and see how well it works.

3. Image generation

- Image generation (synthesis) is the task of generating new images from an existing dataset.
- https://paperswithcode.com/task/image-generation
- Possible projects
  - Pick a public data set and train an image-to-image GAN on it (not as easy as it sounds) or maybe explore diffusion models (easier to train, but harder to implement).
  - Faces are popular (lots of public data sets)
  - You can use generative models to swap faces (i.e., create deep fakes), change emotional expressions in faces, change appearance of images in general, dream up images of non-existing people, and much more.
  - Convert GTA 5 to Cityscapes like here or here.

4. Pose estimation

- Pose Estimation is a general problem in computer vision where we detect the position and orientation of an object.
- Could be the joints of a human (look for 3D Human Pose Estimation) or keypoints on an object (look for Keypoint Detection).
- https://paperswithcode.com/task/pose-estimation
- Possible projects
  - Person tracking in sports (such as golf swing analysis)
  - Facial emotion recognition
  - Object pose estimation using synthetic (i.e., rendered) images
- Data sets / inspiration
  - https://medium.com/datadriveninvestor/3d-pose-estimation-datasets-cd786e50491
  - https://www.di.ens.fr/willow/research/surreal/
  - https://medium.com/@laanlabs/real-time-3d-car-pose-estimation-trained-on-synthetic-data-5fa4a2c16634

Advanced topics

- Few-shot learning: https://paperswithcode.com/task/few-shot-learning
- Visualizing and understanding ConvNets (covered in week 10):
  - https://towardsdatascience.com/understanding-your-convolution-network-with-visualizations-a4883441533b
  - https://ai.googleblog.com/2015/06/inceptionism-going-deeper-into-neural.html
  - https://towardsdatascience.com/visual-interpretability-for-convolutional-neural-networks-2453856210ce
- Multi-task learning (covered in week 6)
  - Train ConvNet on multiple tasks at the same time to increase robustness and make it generalize better.
  - Simultaneous tasks could be 1) Image classification, 2) Object detection, 3) Object segmentation, 4) Pose estimation, and 5) Depth estimation.
- Self-supervised learning (covered in week 13): https://www.youtube.com/watch?v=lgVwtTof1ew
- Analyse video
  - Look for Action Recognition (lots of public dataset) or maybe sign language recognition
- Self-attention model

Some public computer vision datasets

- Meta Pointer: A large collection organized by CV Datasets.
- Yet another Meta pointer
- ImageNet: a large-scale image dataset for visual recognition organized by WordNet hierarchy
- SUN Database: a benchmark for scene recognition and object detection with annotated scene categories and segmented objects
- Places Database: a scene-centric database with 205 scene categories and 2.5 millions of labelled images
- NYU Depth Dataset v2: a RGB-D dataset of segmented indoor scenes
- Microsoft COCO: a new benchmark for image recognition, segmentation and captioning.
- Flickr100M: 100 million creative commons Flickr images
- Labeled Faces in the Wild: a dataset of 13,000 labeled face photographs
- Human Pose Dataset: a benchmark for articulated human pose estimation
- YouTube Faces DB: a face video dataset for unconstrained face recognition in videos
- UCF101: an action recognition data set of realistic action videos with 101 action categories
- Moments in Time: A dataset of one million 3-second videos

Some public computer vision datasets

- https://lionbridge.ai/datasets/20-best-image-datasets-for-computer-vision/
- Labelme: A large dataset created by the MIT Computer Science and Artificial Intelligence Laboratory (CSAIL) containing 187,240 images, 62,197 annotated images, and 658,992 labeled objects.
- Youtube-8M: a large-scale labeled dataset that consists of millions of YouTube video IDs, with annotations of over 3,800+ visual entities.
- Labelled Faces in the Wild: 13,000 labeled images of human faces, for use in developing applications that involve facial recognition.
- Stanford Dogs Dataset: Contains 20,580 images and 120 different dog breed categories, with about 150 images per class.
- Places: Scene-centric database with 205 scene categories and 2.5 million images with a category label.
- CelebFaces: Face dataset with more than 200,000 celebrity images, each with 40 attribute annotations.
- CIFAR-10: A large image dataset of 60,000 32×32 colour images split into 10 classes. The dataset is divided into five training batches and one test batch, each containing 10,000 images.
- Indoor Scene Recognition: A very specific dataset, useful as most scene recognition models are better ‘outside’. Contains 67 Indoor categories, and a total of 15620 images.
- HMDB-51: a large human motion dataset of 51 action classes
- ActivityNet: A large-scale video dataset for human activity understanding