DS-GA.3001 Embodied Learning and Vision

Mengye Ren

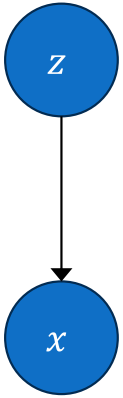
NYU

Spring 2025

embodied-learning-vision-course.github.io

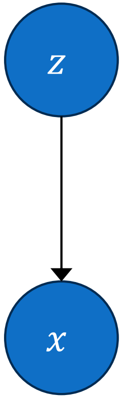
Why Do We Need Learning in Real-World Agents?

- Opinion 1: We always need learning in exploring new environments. There will always be something unknown. There will always be room for improvement. There won't be enough capacity to store all existing knowledge.



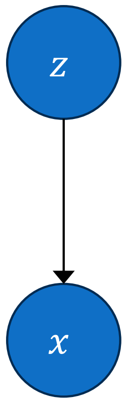
Why Do We Need Learning in Real-World Agents?

- Opinion 1: We always need learning in exploring new environments. There will always be something unknown. There will always be room for improvement. There won't be enough capacity to store all existing knowledge.
- Opinion 2: You can represent infinite variations with finite length description of an abstract symbolic system. We may not have seen all possible variations, but the underlying system remains the same.



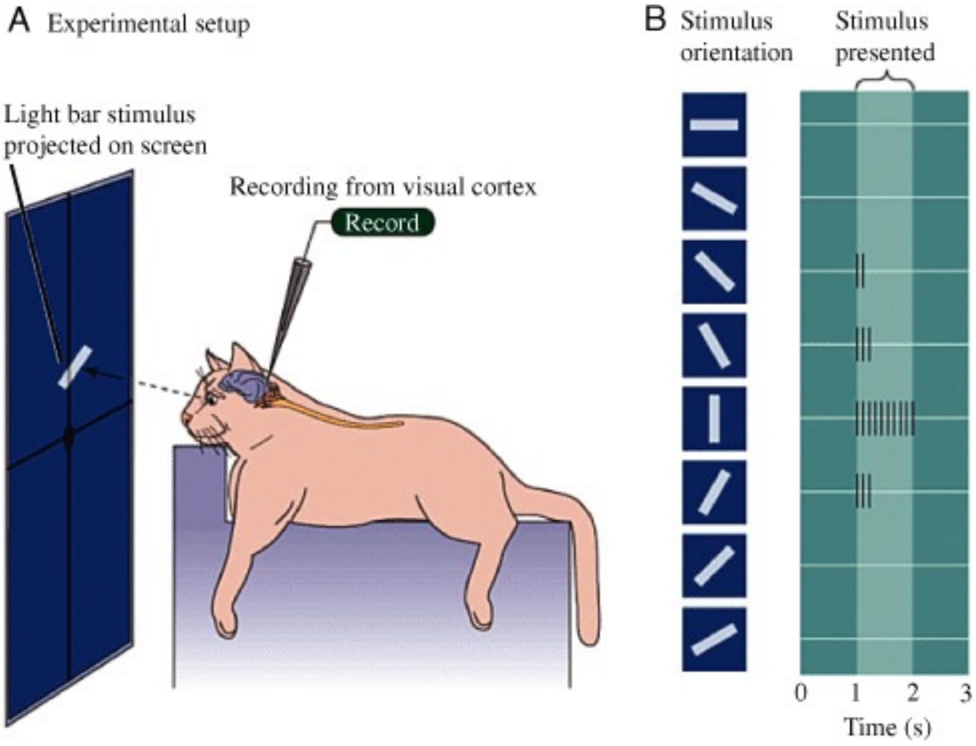
Why Do We Need Learning in Real-World Agents?

- Opinion 1: We always need learning in exploring new environments. There will always be something unknown. There will always be room for improvement. There won't be enough capacity to store all existing knowledge.
- Opinion 2: You can represent infinite variations with finite length description of an abstract symbolic system. We may not have seen all possible variations, but the underlying system remains the same.
- Opinion 3: While theoretically O2 might be true, empirically it is hard to realize. Given limited resource, you might be able to learn more abstract and invariant representations by compressing raw data. You can either be good at one thing without learning, or you need learning to be good at everything.

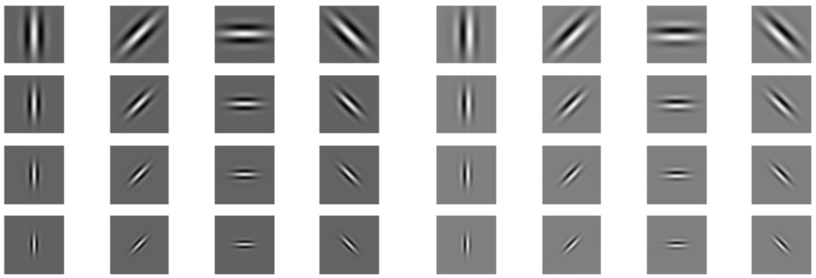


NYU

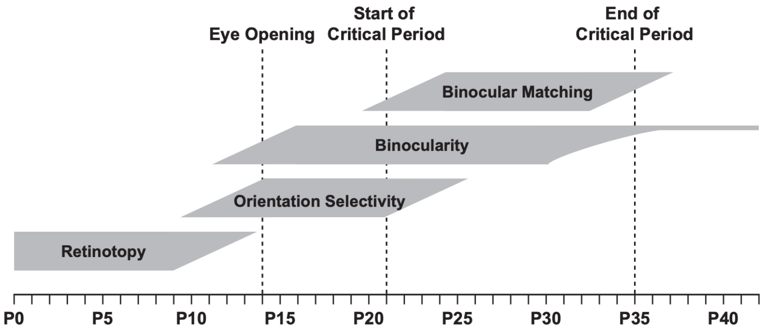
Hubel and Wiesel's Experiments



Wiesel, T.N., and Hubel, D.H. (1963)



Simple Cells, Gabor Filters



Espinosa and Stryker (2012)

Human Developmental Periods

Sensorimotor learning

Simple Reflexes (birth-1 month)

Infants use reflexes such as rooting, sucking, following moving objects with the eyes, and grasping objects. (For example: Infant closes their hand when a toys touches their palm.)

Primary Circular Reactions (1-4 months)

A primary circular reaction is when an infant tries to reproduce an event that happened by accident because they find it to be pleasurable. (For example: Intentionally mouthing a toy bunny.)

Secondary Circular Reactions (4-8 months)

Child becomes more focused on the world and begins to intentionally repeat an action in order to trigger an environmental response. (For example: purposefully picking up a pacifier to put it in their mouth.)

Coordination Of Secondary Circular Reactions (8-12 months)

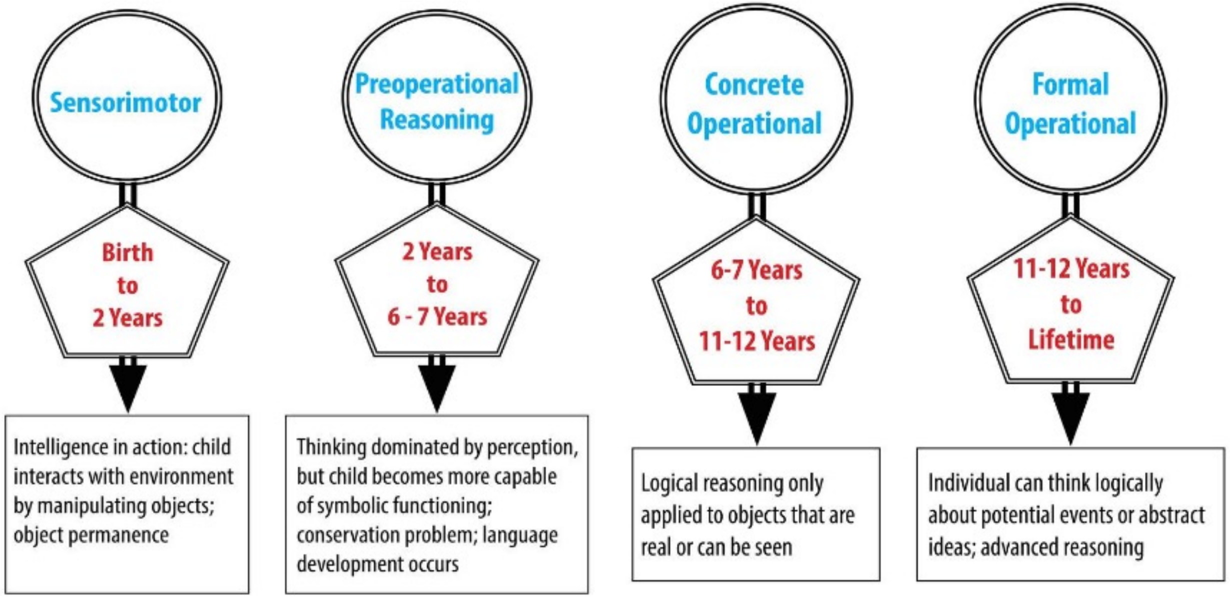
Child acts intentionally and follows steps to achieve goals. Child begin to do things intentionally and understands object permanence. (For example: Child will push one toy aside to get to a second toy partially concealed underneath.)

Tertiary Circular Reactions (12-18 months)

Child discovers new means to meet goals and begins to modify earlier behaviors to meet existing needs. Piaget described children in this stage as “young scientists”. (For example: Child repeatedly drops/throws a set of plastic keys and observes how they move through space.)

Internalization of schemas (18-24 months)

Child begins to use symbols and form mental representations. The beginnings of insight and creativity are associated with this stage. (For example: Child pushes a chair across the kitchen and climbs up on it to reach a cookie on the counter.)

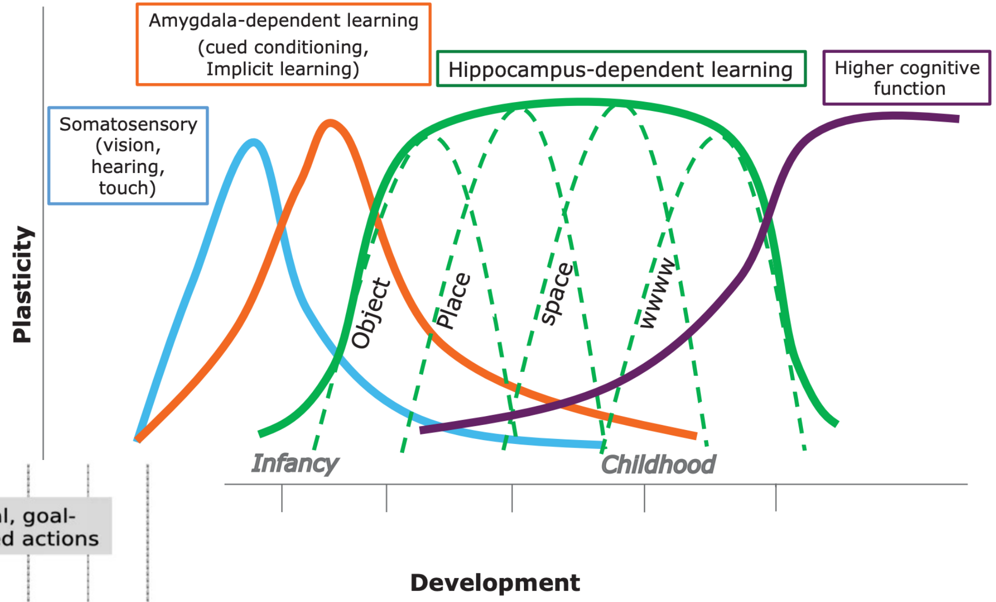


Piaget's Theory of Cognitive Development

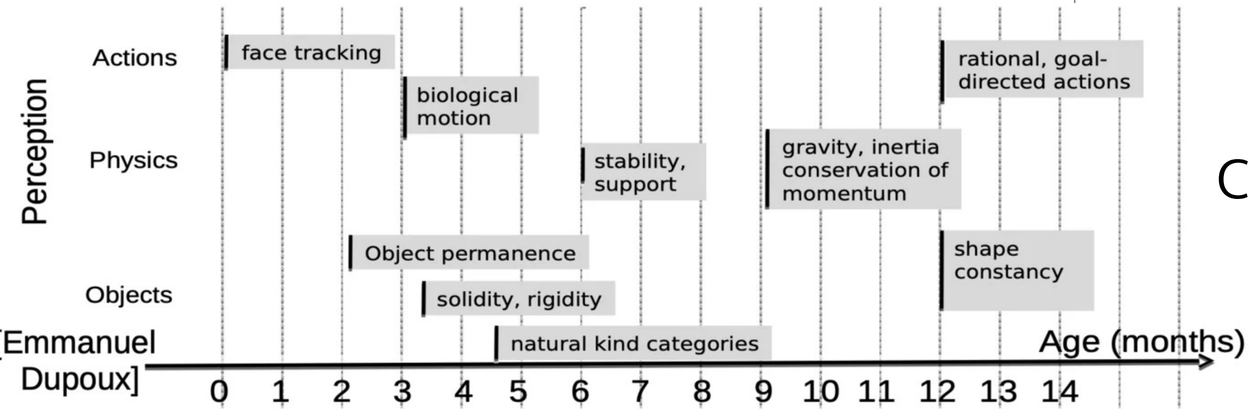
NYU

Insights from the Brain

- Perception and motion



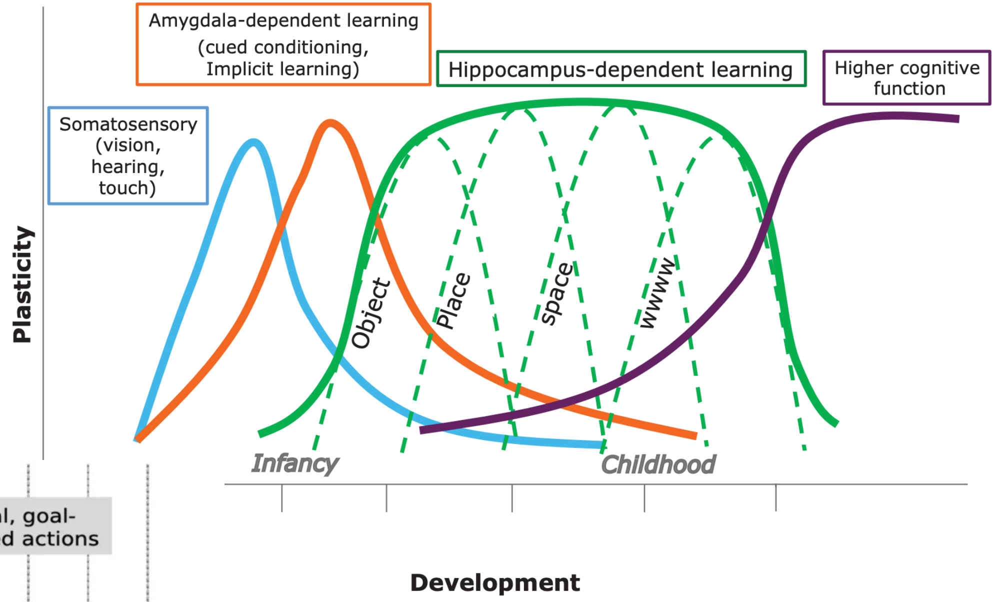
Critical Periods of Development



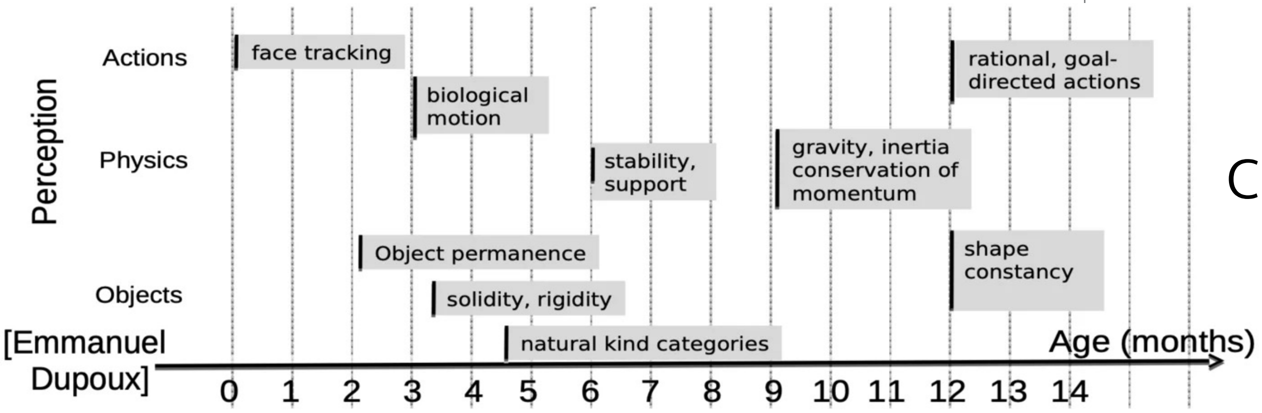
[Emmanuel Dupoux]

Insights from the Brain

- Perception and motion
- Low-level to high-level representation

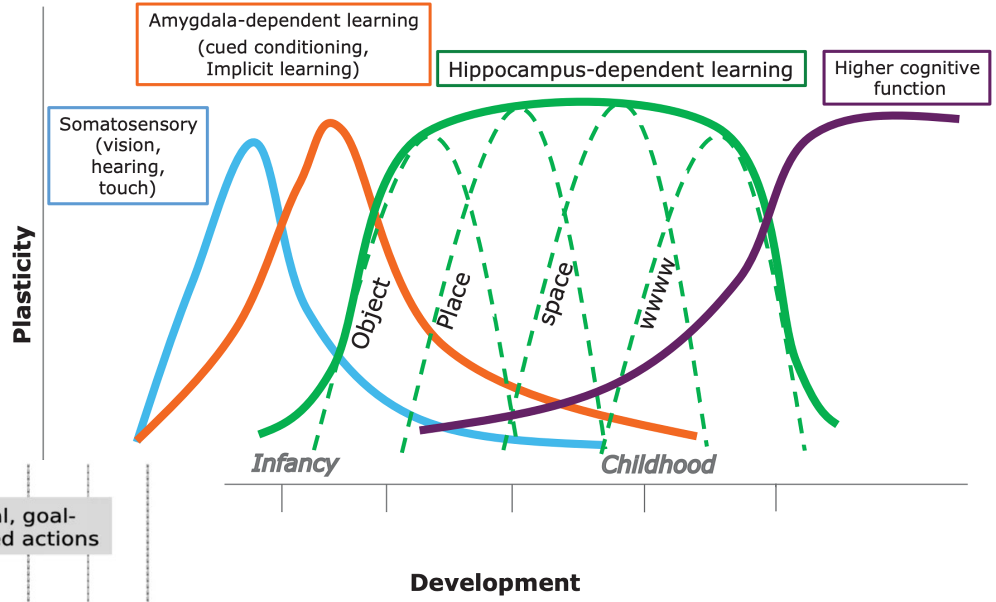


Critical Periods of Development

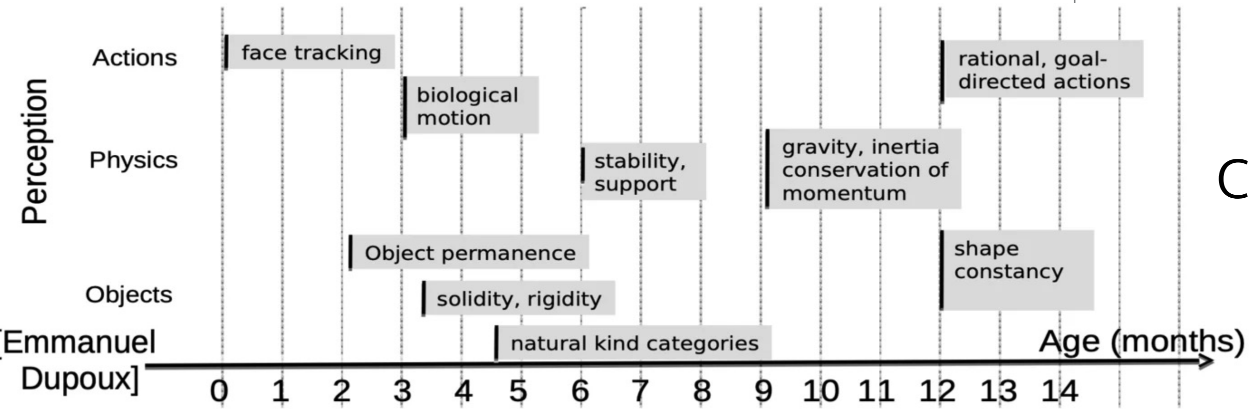


Insights from the Brain

- Perception and motion
- Low-level to high-level representation
- Hippocampus and memory



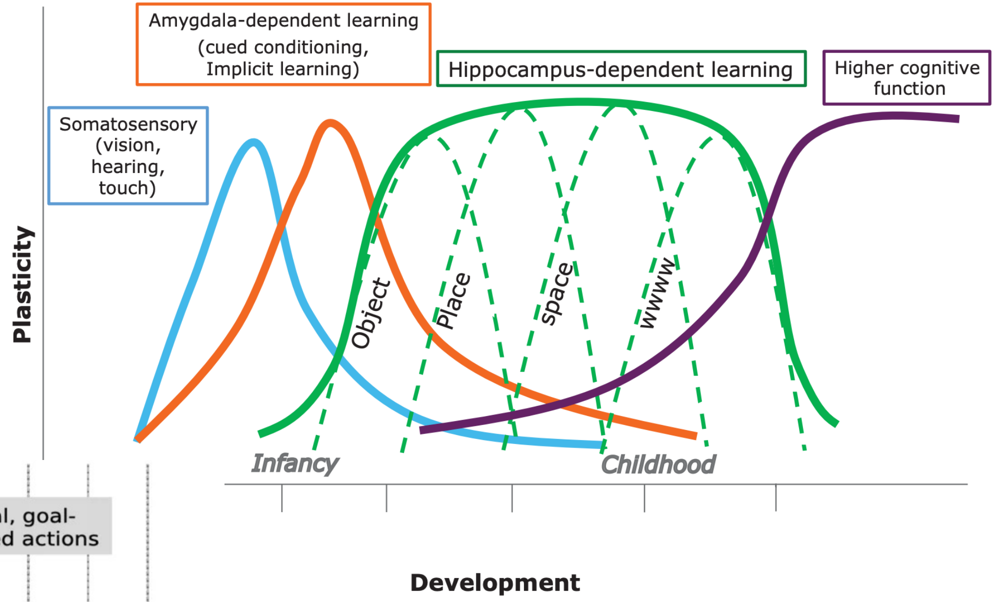
Critical Periods of Development



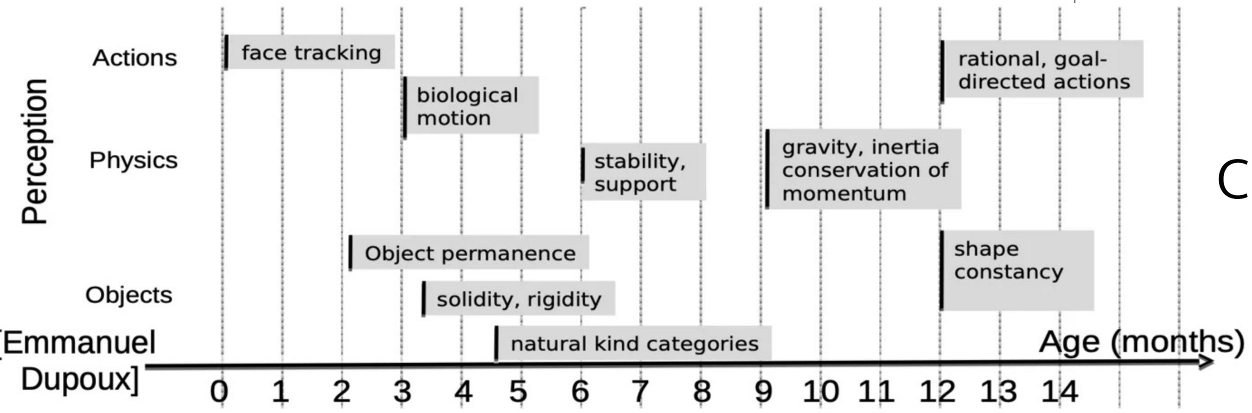
[Emmanuel Dupoux]

Insights from the Brain

- Perception and motion
- Low-level to high-level representation
- Hippocampus and memory
- Abstraction



Critical Periods of Development



[Emmanuel Dupoux]

NYU

Direction 1: End-to-end self-supervised learning for perception and planning

- Existing end-to-end learning-based planning frameworks are mostly focused on supervised learning of labeled objects and human demonstrations.

Direction 1: End-to-end self-supervised learning for perception and planning

- Existing end-to-end learning-based planning frameworks are mostly focused on supervised learning of labeled objects and human demonstrations.
- Demonstrations and rewards are forms of labels.

Direction 1: End-to-end self-supervised learning for perception and planning

- Existing end-to-end learning-based planning frameworks are mostly focused on supervised learning of labeled objects and human demonstrations.
- Demonstrations and rewards are forms of labels.
- How do we achieve label-efficient learning and exploration through self-supervision?

Direction 1: End-to-end self-supervised learning for perception and planning

- Existing end-to-end learning-based planning frameworks are mostly focused on supervised learning of labeled objects and human demonstrations.
- Demonstrations and rewards are forms of labels.
- How do we achieve label-efficient learning and exploration through self-supervision?
- Is planning and action necessary for a label-efficient algorithm for perception?

NYU

Direction 1: End-to-end self-supervised learning for perception and planning

- Existing end-to-end learning-based planning frameworks are mostly focused on supervised learning of labeled objects and human demonstrations.
- Demonstrations and rewards are forms of labels.
- How do we achieve label-efficient learning and exploration through self-supervision?
- Is planning and action necessary for a label-efficient algorithm for perception?
- Explore the full spectrum from end-to-end learning to modular designs.

NYU

Direction 2: Enhancing foundation models for spatial intelligence

- Foundation model are trained with discrete tokens and are less familiar with the 3D world to perform exact perception, inference and planning.

Direction 2: Enhancing foundation models for spatial intelligence

- Foundation model are trained with discrete tokens and are less familiar with the 3D world to perform exact perception, inference and planning.
- Augment pretrained foundation models with the ability to perceive and plan under precision in embodied environments.

Direction 2: Enhancing foundation models for spatial intelligence

- Foundation model are trained with discrete tokens and are less familiar with the 3D world to perform exact perception, inference and planning.
- Augment pretrained foundation models with the ability to perceive and plan under precision in embodied environments.
- How do we enhance robustness in real-world environments?

Direction 2: Enhancing foundation models for spatial intelligence

- Foundation model are trained with discrete tokens and are less familiar with the 3D world to perform exact perception, inference and planning.
- Augment pretrained foundation models with the ability to perceive and plan under precision in embodied environments.
- How do we enhance robustness in real-world environments?
- Can be synthetic/realistic, 2D/3D environments.

NYU

Direction 2: Enhancing foundation models for spatial intelligence

- Foundation model are trained with discrete tokens and are less familiar with the 3D world to perform exact perception, inference and planning.
- Augment pretrained foundation models with the ability to perceive and plan under precision in embodied environments.
- How do we enhance robustness in real-world environments?
- Can be synthetic/realistic, 2D/3D environments.
- Can models with geometric designs beat generic foundation models in terms of learning efficiency?

NYU

Direction 3: Continual learning for embodied intelligence

- How do we apply continual learning algorithms to embodied tasks?

NYU

Direction 3: Continual learning for embodied intelligence

- How do we apply continual learning algorithms to embodied tasks?
- Skill learning, open world learning

Direction 3: Continual learning for embodied intelligence

- How do we apply continual learning algorithms to embodied tasks?
- Skill learning, open world learning
- Memory design, retrieval augmentation, continuous finetuning

Direction 3: Continual learning for embodied intelligence

- How do we apply continual learning algorithms to embodied tasks?
- Skill learning, open world learning
- Memory design, retrieval augmentation, continuous finetuning
- Incremental learning with experience/action abstraction

NYU

Direction 3: Continual learning for embodied intelligence

- How do we apply continual learning algorithms to embodied tasks?
- Skill learning, open world learning
- Memory design, retrieval augmentation, continuous finetuning
- Incremental learning with experience/action abstraction
- Replay with physical constraints

Direction 3: Continual learning for embodied intelligence

- How do we apply continual learning algorithms to embodied tasks?
- Skill learning, open world learning
- Memory design, retrieval augmentation, continuous finetuning
- Incremental learning with experience/action abstraction
- Replay with physical constraints
- Actively choosing learning objectives

NYU

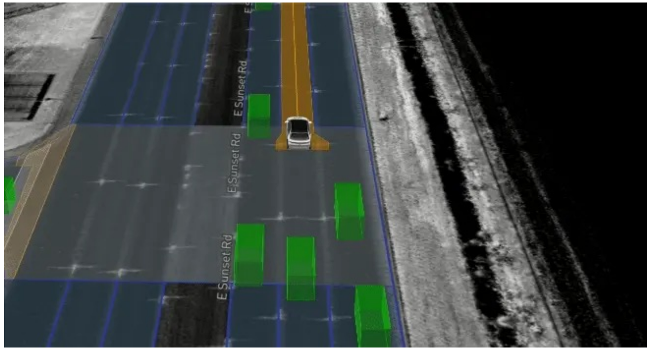
Other Directions?

- You are allowed to form your own research ideas.
- Need to get my approval first. Talk to me early in the semester.

Embodied Environments

- You **must** demonstrate your project in an embodied environment.

Habitat indoor home



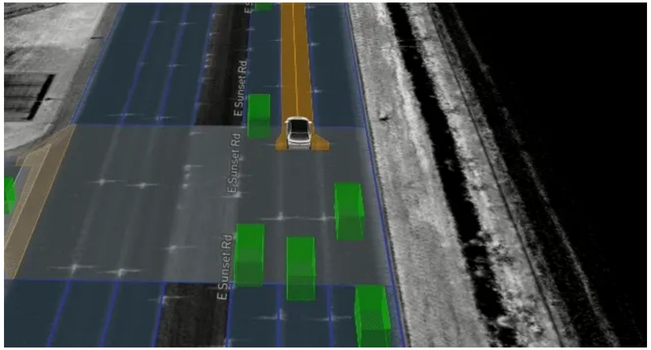
NuPlan self-driving

Ego-Exo4D Egocentric Videos

Embodied Environments

- You **must** demonstrate your project in an embodied environment.
- You can focus on one aspect of the algorithm. No need for a full stack.

Habitat indoor home



NuPlan self-driving

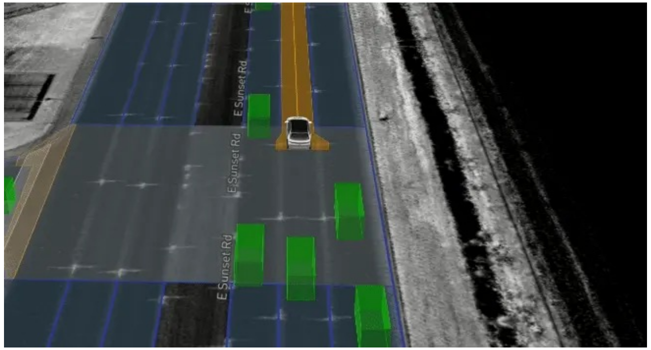
Ego-Exo4D Egocentric Videos

NYU

Embodied Environments

- You **must** demonstrate your project in an embodied environment.
- You can focus on one aspect of the algorithm. No need for a full stack.
- Your TAs will showcase demos on some exemplar environments.

Habitat indoor home



NuPlan self-driving

Ego-Exo4D Egocentric Videos

GenAI Policy

- AI may not be used in weekly paper reviews and paper presentations (except AI illustrations).

GenAI Policy

- AI may not be used in weekly paper reviews and paper presentations (except AI illustrations).
- AI may be used towards coding assistance and report writing assistance in the course project.

GenAI Policy

- AI may not be used in weekly paper reviews and paper presentations (except AI illustrations).
- AI may be used towards coding assistance and report writing assistance in the course project.
- The use of AI can still impact the grade if the report contains poor writings and non-factual statements.

Office Hours

- Myself: Thursday 1:00pm – 2:00pm Zoom Link on course website and calendar.
  - In person by appointment Room 508, 60 5th Ave

Office Hours

- Myself: Thursday 1:00pm – 2:00pm Zoom Link on course website and calendar.
  - In person by appointment Room 508, 60 5th Ave
- TAs:

Chris Hoang
Wed 2-3PM
Room 502

Ying Wang
Thu 2-3PM
Room 763

Paper Review

- Week 2 due next Thursday
- Week 3 due on the same day as W2
- Choose from the recent papers (<= 3 years)

Content

- Introduction and Brief History

Content

- Introduction and Brief History
- Deep Learning and Structured Outputs

Content

- Introduction and Brief History
- Deep Learning and Structured Outputs
- 3D Vision and Mapping

Content

- Introduction and Brief History
- Deep Learning and Structured Outputs
- 3D Vision and Mapping
- Self-Supervised Representation Learning and Object Discovery

Content

- Introduction and Brief History
- Deep Learning and Structured Outputs
- 3D Vision and Mapping
- Self-Supervised Representation Learning and Object Discovery
- World Models and Forecasting

NYU

Content

- Introduction and Brief History
- Deep Learning and Structured Outputs
- 3D Vision and Mapping
- Self-Supervised Representation Learning and Object Discovery
- World Models and Forecasting
- End-to-End Planning

NYU

Content

- Introduction and Brief History
- Deep Learning and Structured Outputs
- 3D Vision and Mapping
- Self-Supervised Representation Learning and Object Discovery
- World Models and Forecasting
- End-to-End Planning
- Continual Learning

Content

- Introduction and Brief History
- Deep Learning and Structured Outputs
- 3D Vision and Mapping
- Self-Supervised Representation Learning and Object Discovery
- World Models and Forecasting
- End-to-End Planning
- Continual Learning
- Few-Shot Learning

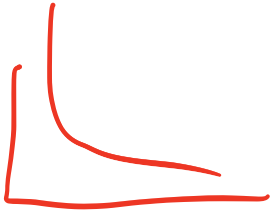
Content

- Introduction and Brief History
- Deep Learning and Structured Outputs
- 3D Vision and Mapping
- Self-Supervised Representation Learning and Object Discovery
- World Models and Forecasting
- End-to-End Planning
- Continual Learning
- Few-Shot Learning
- LLM Agents

NYU

Deep Learning

- Over decades, optimizing deep neural networks was not trivial.

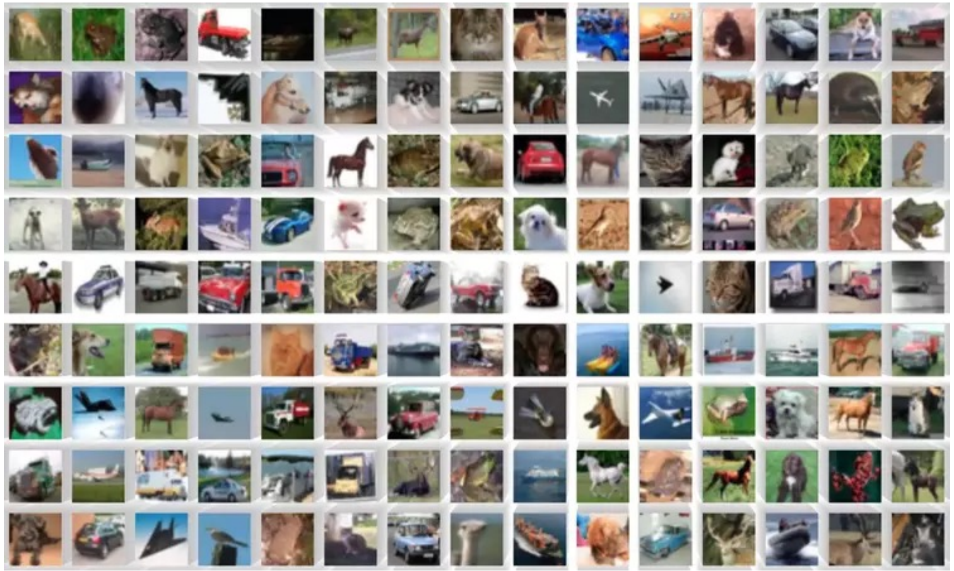


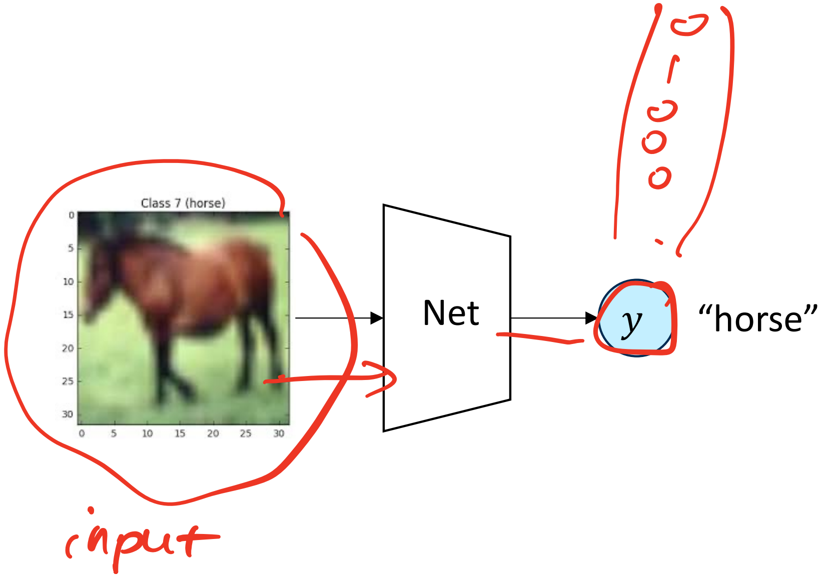
Deep Learning

- Over decades, optimizing deep neural networks was not trivial.
- Progress came from (taken for granted nowadays):
  - Initialization
  - Normalization (BN, LN, GN, etc.)
  - Skip connection (recurrent net, residual net)
  - Regularization (dropout, noise, augmentation)
  - Attention (generalization)

Classification

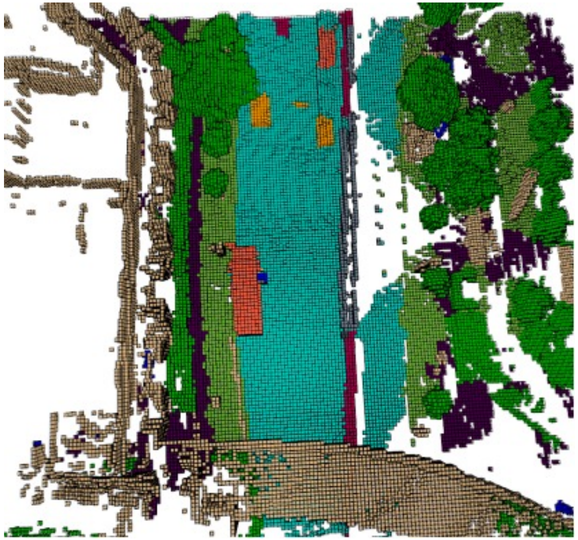
- To test how we can fit a deep neural network well, people have relied on simple benchmarks, such as image classification.





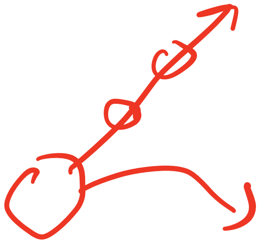
What are Structured Outputs?

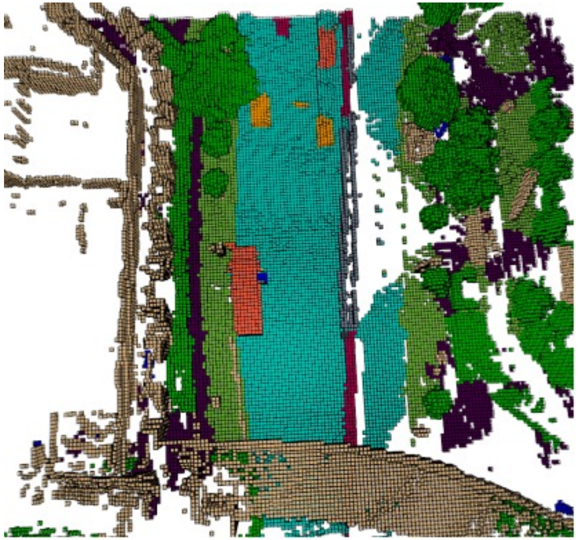
- An embodied agent needs to have a structured output space.
  - Object localization, tracking, motion, spatial segmentation, 3d occupancy
  - Trajectory, forecasting, planning



What are Structured Outputs?

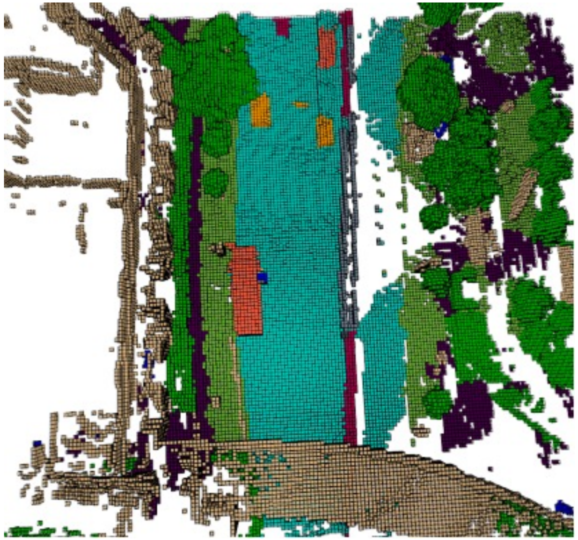
- An embodied agent needs to have a structured output space.
  - Object localization, tracking, motion, spatial segmentation, 3d occupancy
  - Trajectory, forecasting, planning
- A naïve solution is to simply have multiple output dimensions.





What are Structured Outputs?

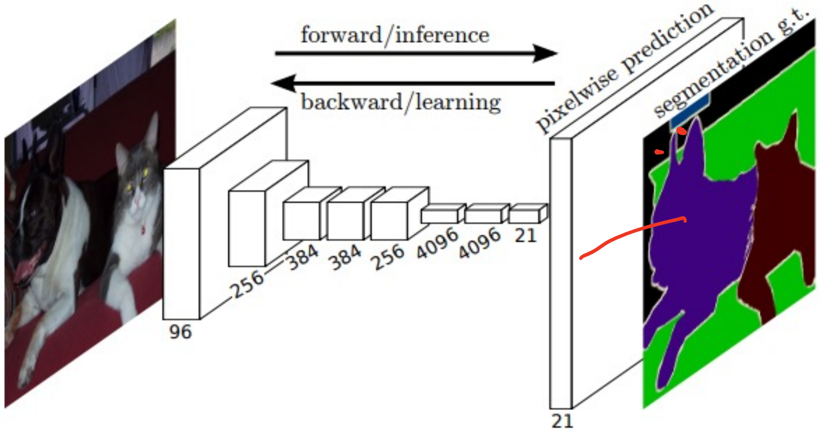
- An embodied agent needs to have a structured output space.
  - Object localization, tracking, motion, spatial segmentation, 3d occupancy
  - Trajectory, forecasting, planning
- A naïve solution is to simply have multiple output dimensions.
- It often does not reason the joint probability



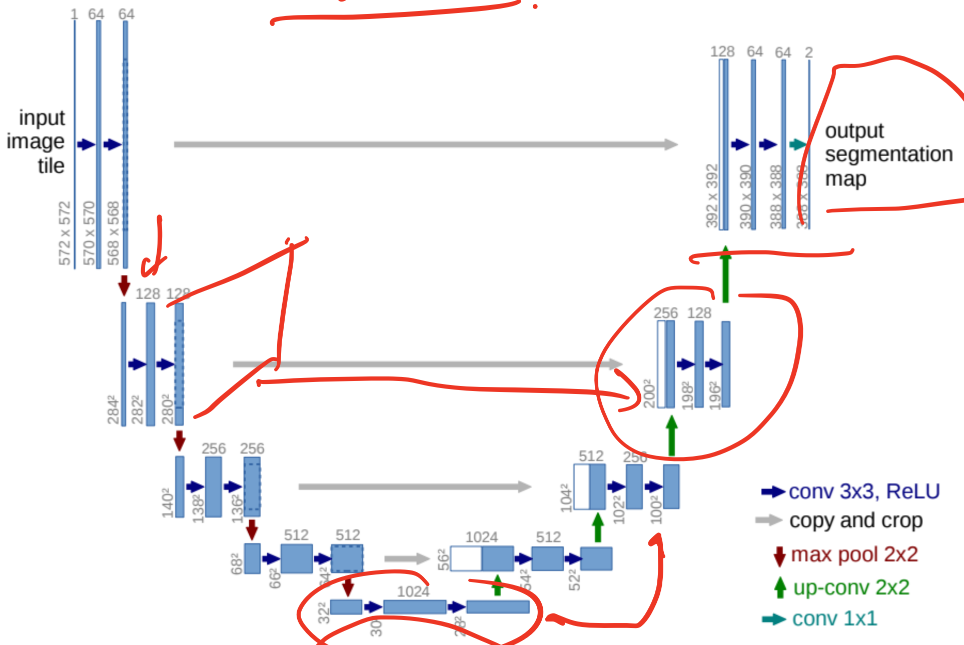
NYU

Network Architecture for Structured Outputs

- Segmentation
- Spatial, high-resolution



UNet

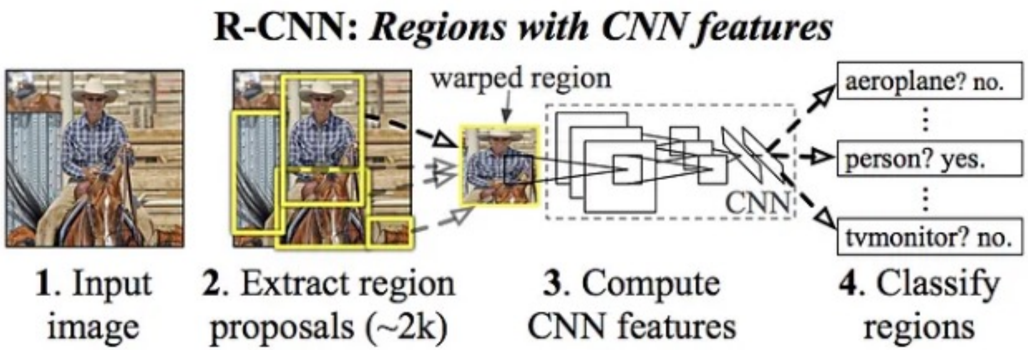


global

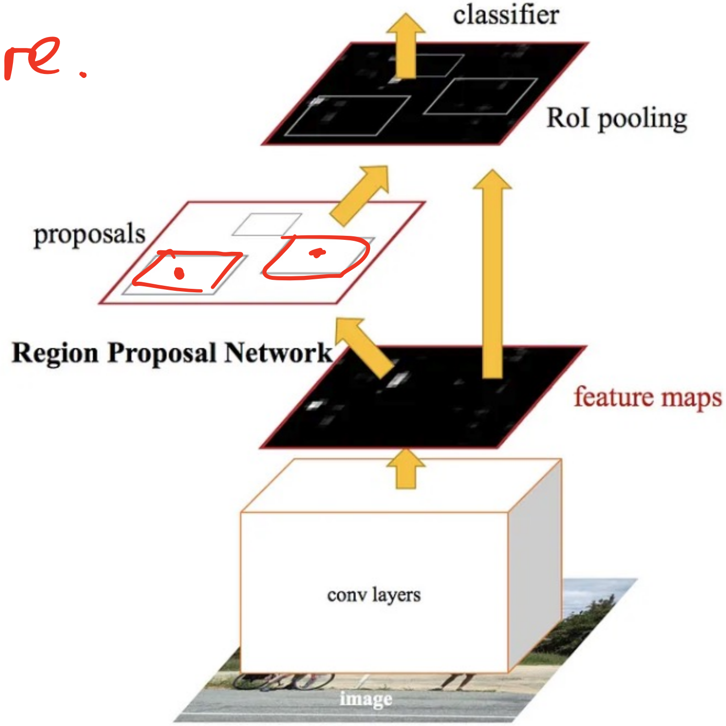
Network Architecture for Structured Outputs

proposal → refined.
→ score.

- Object Detection



Girshick et al., 2013

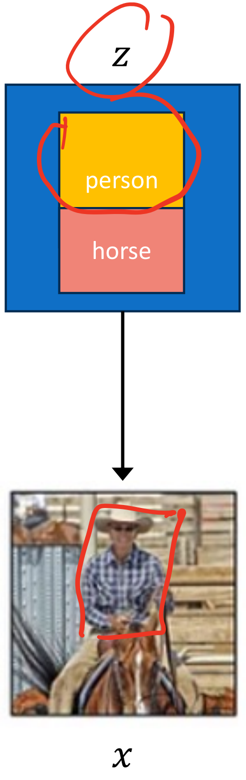


Ren et al., 2015

NYU

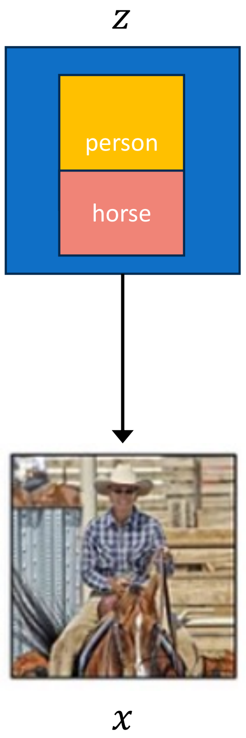
Object Detection as Inference

- Bounding boxes are structured latent variables.



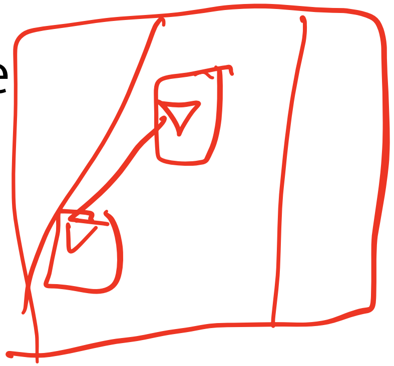
Object Detection as Inference

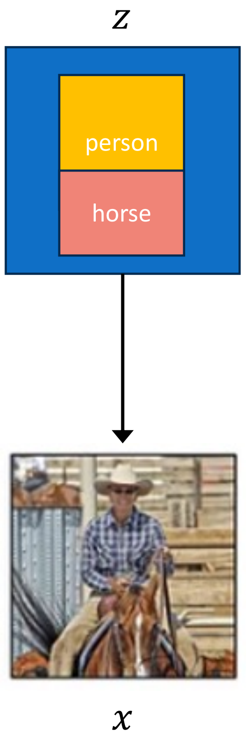
- Bounding boxes are structured latent variables.
- Occupancy as physical constraints.
  - One spatial 3D location can only present a single physical object.



Object Detection as Inference

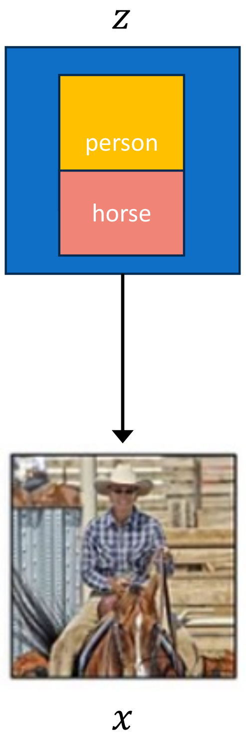
- Bounding boxes are structured latent variables.
- Occupancy as physical constraints.
  - One spatial 3D location can only present a single physical object.
- Object co-occurrence in the scene
  - Context





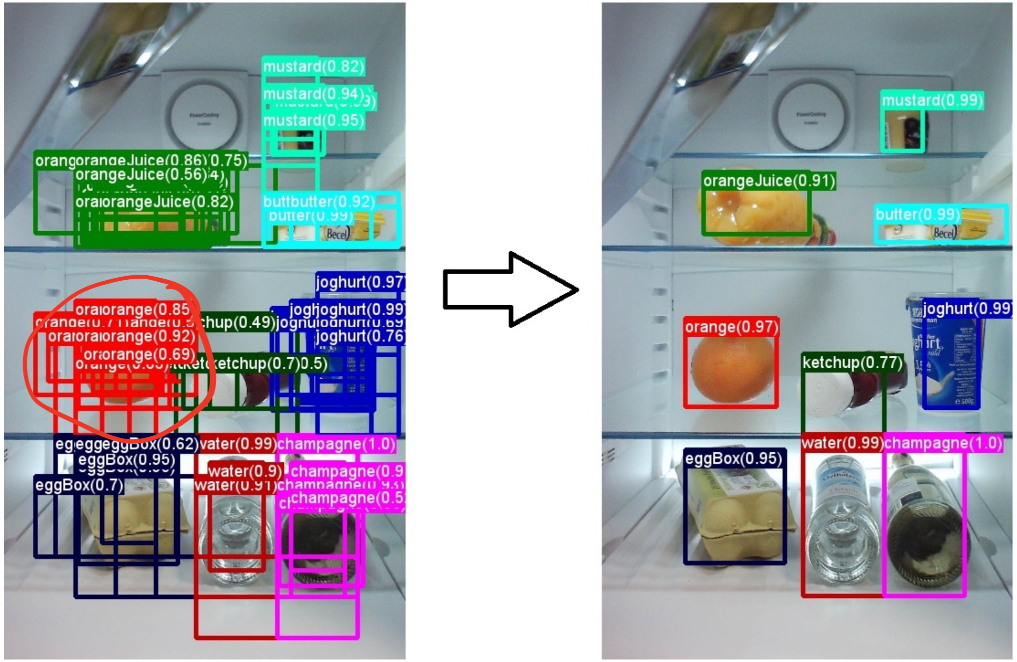
Object Detection as Inference

- Bounding boxes are structured latent variables.
- Occupancy as physical constraints.
  - One spatial 3D location can only present a single physical object.
- Object co-occurrence in the scene
  - Context
- The role of the network is to perform “inference” on the latent variables.



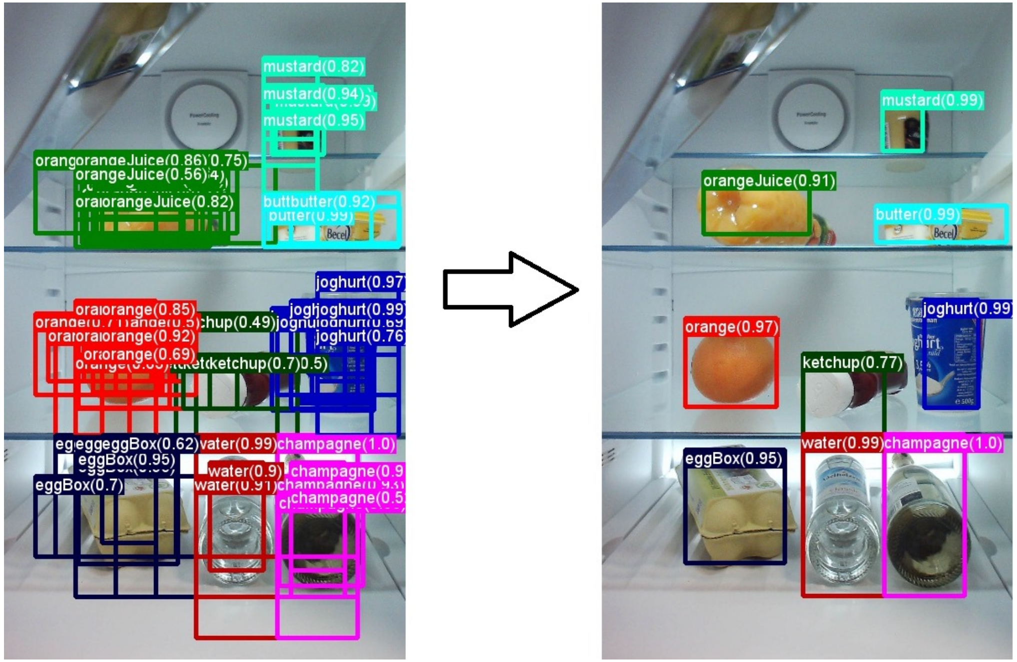
Non-Maximal Suppression

- Box Proposals: Samples of boxes that may come from a single object (latent)



Non-Maximal Suppression

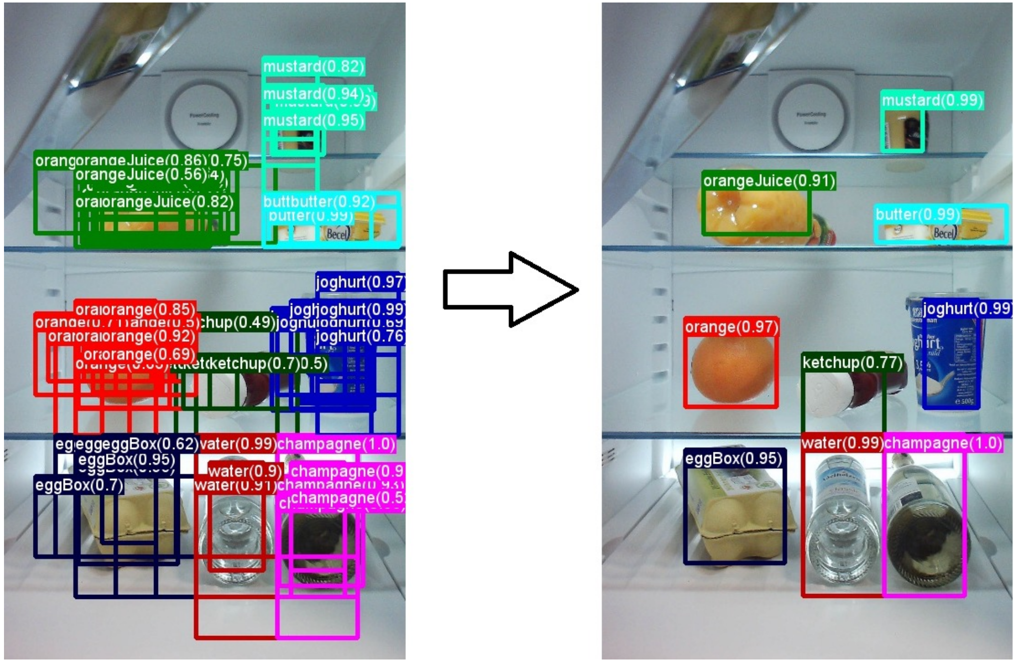
- Box Proposals: Samples of boxes that may come from a single object (latent)
- Each with a confidence score, density representation of the inferred latent distribution $q(z|x)$.



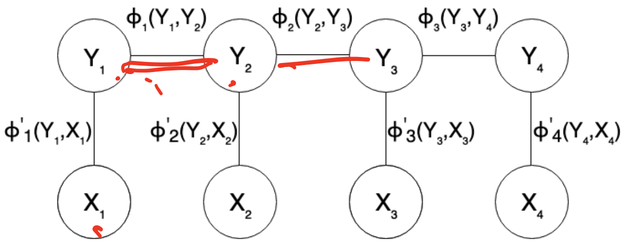
NYU

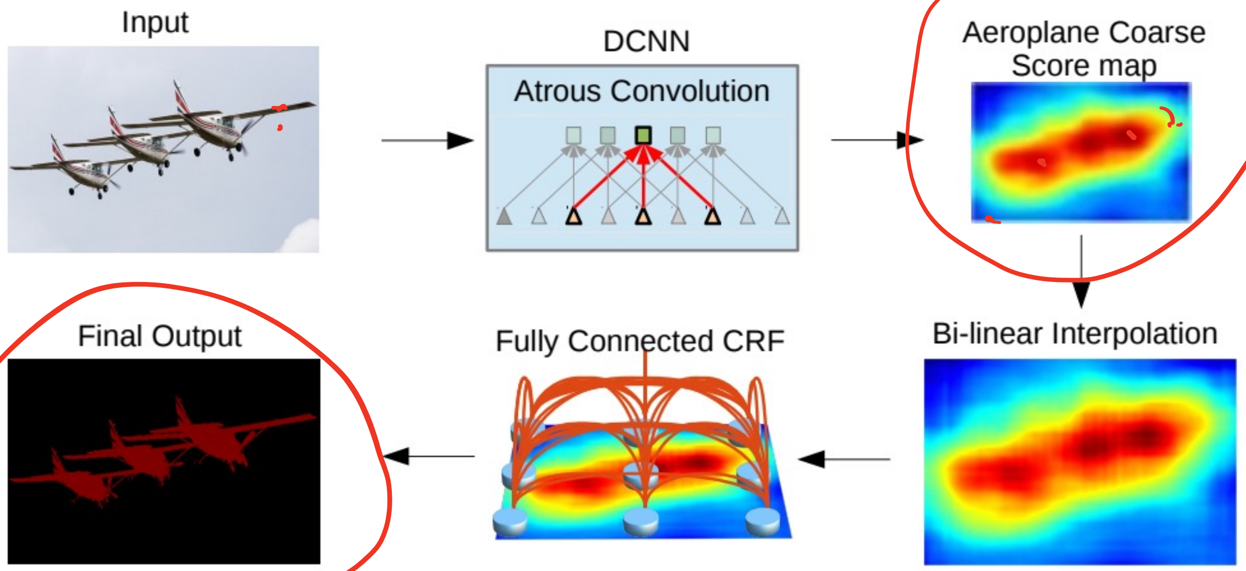
Non-Maximal Suppression

- Box Proposals: Samples of boxes that may come from a single object (latent)
- Each with a confidence score, density representation of the inferred latent distribution $q(z|x)$.
- MAP: take the mode of the distribution



Segmentation as CRF Inference





Chen et al., 2015

Inference Problem

- Knows $p(x|z)$.

Inference Problem

$$\begin{tikzcd} z \arrow[d] \\ x \arrow[u, bend right] \end{tikzcd}$$

- Knows $p(x|z)$.
- Wants to know $p(z|x)$. Bayes rule.

$$p(z|x;\theta) = \frac{p(x,z;\theta)}{\int_z p(x,z;\theta)}.$$

Inference Problem

- Knows $p(x|z)$.
- Wants to know $p(z|x)$. Bayes rule.

$$p(z|x;\theta) = \frac{p(x,z;\theta)}{\int_z p(x,z;\theta)}.$$

- Brute force

Inference Problem

- Knows $p(x|z)$.
- Wants to know $p(z|x)$. Bayes rule.

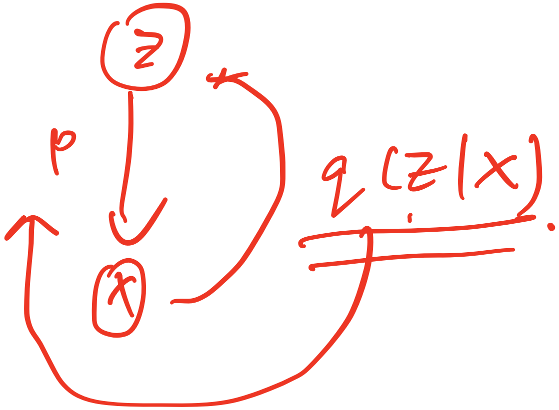
$$p(z|x;\theta) = \frac{p(x,z;\theta)}{\int_z p(x,z;\theta)}.$$

- Brute force
- Message passing, sum-product, belief propagation (DP)

Inference Problem

- Knows $p(x|z)$.
- Wants to know $p(z|x)$. Bayes rule.

$$p(z|x;\theta) = \frac{p(x,z;\theta)}{\int_z p(x,z;\theta)}.$$



- Brute force
- Message passing, sum-product, belief propagation (DP)
- Variational inference, mean field

Inference Problem

- Knows $p(x|z)$.
- Wants to know $p(z|x)$. Bayes rule.

$$p(z|x;\theta) = \frac{p(x,z;\theta)}{\int_z p(x,z;\theta)}.$$

- Brute force
- Message passing, sum-product, belief propagation (DP)
- Variational inference, mean field
- Stochastic sampling, MCMC

Variational Inference

- Jensens inequality to get ELBO.

$$\log p(x) = \log \int_z p(x,z)$$

$$= \log \int_z p(x,z) \frac{q(z)}{q(z)}$$

Variational Inference

- Jensens inequality to get ELBO.

$$\log p(x) = \log \int_z p(x,z)$$
$$= \log \int_z p(x,z) \frac{q(z)}{q(z)}$$
$$= \log \left(\mathbb{E}_q \frac{p(x,z)}{q(z)} \right)$$

NYU

Variational Inference

- Jensens inequality to get ELBO.

$$
\begin{aligned}
\log p(x) &= \log \int_z p(x,z) \\
&= \log \int_z p(x,z) \frac{q(z)}{q(z)} \\
&= \log \left(\mathbb{E}_q \frac{p(x,z)}{q(z)} \right) \\
&\geq \mathbb{E}_q \log \frac{p(x,z)}{q(z)} \\
&= \mathbb{E}_q \log p(x,z) - \mathbb{E}_q \log q(z) = \mathcal{L}.
\end{aligned}
$$

Mean-Field Inference

- If there are many latent variables, we can assume factorization (local variational approximation):

$$q(z_1, \ldots, z_m) = \prod_{j=1}^{m} q(z_j).$$

Mean-Field Inference

- If there are many latent variables, we can assume factorization (local variational approximation):

$$q(z_1, \ldots, z_m) = \prod_{j=1}^{m} q(z_j).$$

$$\mathcal{L} = \log p(x) + \sum_{j=1}^{m} \mathbb{E}_{q(z_j)} \log p(z_j | z_{-j}, x) - \mathbb{E}_{q(z_j)} \log(q(z_j)).$$

NYU

Inference Operations

- CRF with pairwise energy. Use x as labels.

$$E(\mathbf{x}) = \underbrace{\sum_i \psi_u(x_i)}_{\text{unary}} + \sum_{i<j} \psi_p(x_i, x_j),$$

[Krähenbühl & Koltun, 2012]

Inference Operations

- CRF with pairwise energy. Use x as labels.

$$E(\mathbf{x}) = \sum_i \psi_u(x_i) + \sum_{i<j} \psi_p(x_i, x_j),$$

$$\psi_p(x_i, x_j) = \mu(x_i, x_j) \underbrace{\sum_{m=1}^{K} w^{(m)} k^{(m)}(\mathbf{f}_i, \mathbf{f}_j)}_{k(\mathbf{f}_i, \mathbf{f}_j)}$$

[Krähenbühl & Koltun, 2012]

NYU

Inference Operations

- CRF with pairwise energy. Use x as labels.

$$E(\mathbf{x}) = \sum_i \psi_u(x_i) + \sum_{i<j} \psi_p(x_i, x_j),$$

pixel

$$\psi_p(x_i, x_j) = \mu(x_i, x_j) \underbrace{\sum_{m=1}^{K} w^{(m)} k^{(m)}(\mathbf{f}_i, \mathbf{f}_j)}_{k(\mathbf{f}_i, \mathbf{f}_j)}$$

location

$$k^{(m)}(\mathbf{f}_i, \mathbf{f}_j) = \exp(-\tfrac{1}{2}(\mathbf{f}_i - \mathbf{f}_j)^{\mathrm{T}} \Lambda^{(m)} (\mathbf{f}_i - \mathbf{f}_j)).$$

$$\underbrace{w^{(1)} \exp\left(-\frac{|p_i - p_j|^2}{2\theta_\alpha^2} - \frac{|I_i - I_j|^2}{2\theta_\beta^2}\right)}_{\text{appearance kernel}} + \underbrace{w^{(2)} \exp\left(-\frac{|p_i - p_j|^2}{2\theta_\gamma^2}\right)}_{\text{smoothness kernel}}.$$

$$\mu(x_i, x_j) = [x_i \neq x_j]$$

bird ≠ sky.

[Krähenbühl & Koltun, 2012]

Inference in Fully Connected CRF

- Iterative mean-field inference.

Algorithm 1 Mean field in fully connected CRFs

Initialize Q $\qquad\triangleright\ Q_i(x_i) \leftarrow \frac{1}{Z_i}\exp\{-\phi_u(x_i)\}$

while not converged **do** $\qquad\triangleright$ See Section 6 for convergence analysis

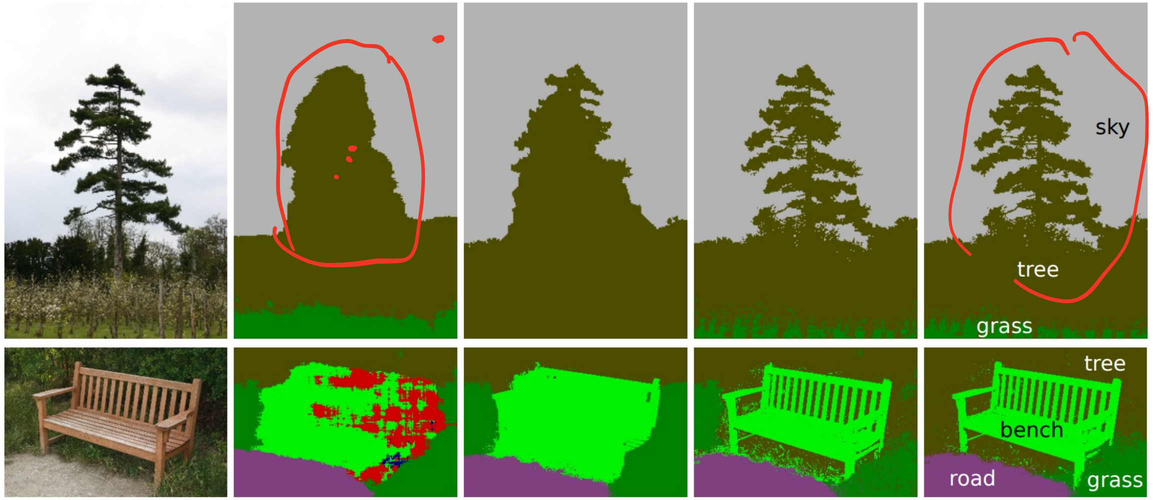
$\quad\tilde{Q}_i^{(m)}(l) \leftarrow \sum_{j\neq i} k^{(m)}(\mathbf{f}_i, \mathbf{f}_j) Q_j(l)$ for all m $\qquad\triangleright$ **Message passing** from all X_j to all X_i

$\quad\hat{Q}_i(x_i) \leftarrow \sum_{l\in\mathcal{L}} \mu^{(m)}(x_i, l) \sum_m w^{(m)} \tilde{Q}_i^{(m)}(l)$ $\qquad\triangleright$ **Compatibility transform**

$\quad Q_i(x_i) \leftarrow \exp\{-\psi_u(x_i) - \hat{Q}_i(x_i)\}$ $\qquad\triangleright$ **Local update**

$\quad$ normalize $Q_i(x_i)$

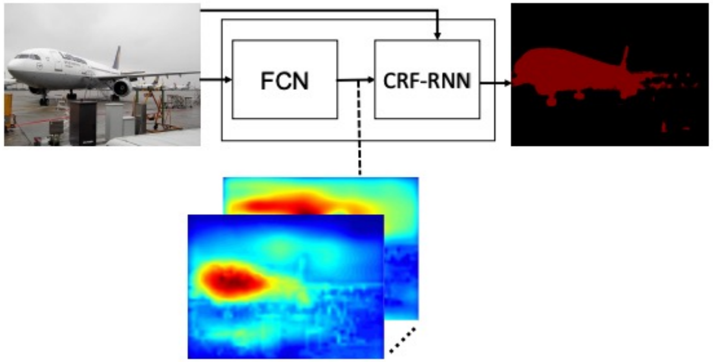
end while



(a) Image (b) Unary classifiers (c) Robust P^n CRF (d) Fully connected CRF, MCMC inference, 36 hrs (e) Fully connected CRF, our approach, 0.2 seconds

[Krähenbühl & Koltun, 2012]

CRFs as RNNs



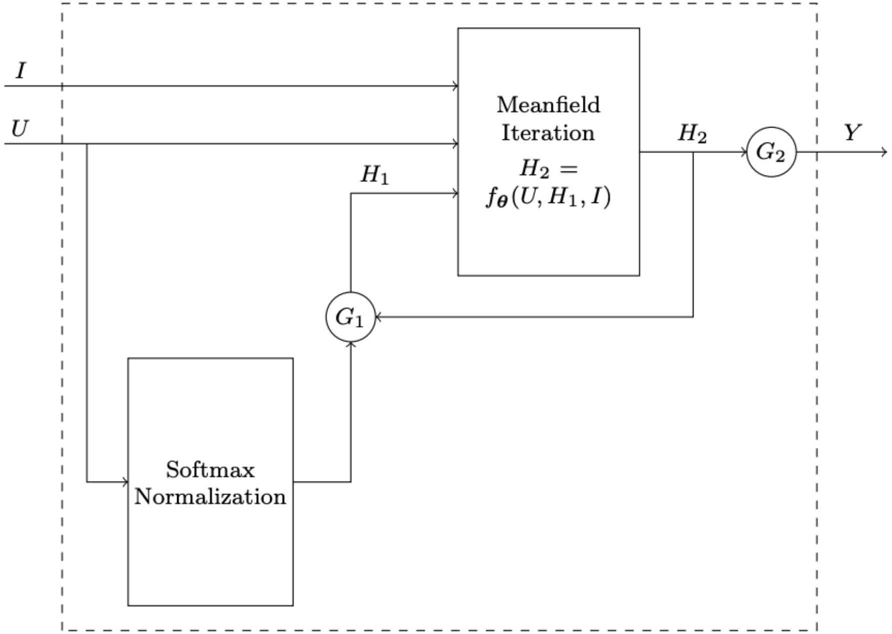


Figure 2. **The CRF-RNN Network.** We formulate the iterat mean-field algorithm as a Recurrent Neural Network (RNN). G ing functions G_1 and G_2 are fixed as described in the text.

$Q_i(l) \leftarrow \frac{1}{Z_i}\exp\left(U_i(l)\right)$ for all i ▷ Initialization

while not converged **do**

$\tilde{Q}_i^{(m)}(l) \leftarrow \sum_{j\neq i} k^{(m)}(\mathbf{f}_i, \mathbf{f}_j)Q_j(l)$ for all m ▷ Message Passing

$\check{Q}_i(l) \leftarrow \sum_m w^{(m)}\tilde{Q}_i^{(m)}(l)$ ▷ Weighting Filter Outputs

$\hat{Q}_i(l) \leftarrow \sum_{l'\in\mathcal{L}} \mu(l, l')\check{Q}_i(l')$ ▷ Compatibility Transform

$\breve{Q}_i(l) \leftarrow U_i(l) - \hat{Q}_i(l)$ ▷ Adding Unary Potentials

$Q_i \leftarrow \frac{1}{Z_i}\exp\left(\breve{Q}_i(l)\right)$ ▷ Normalizing

end while

[Zheng et al., 2015]

NYU

Summary

- Perception of high dimensional objects can be viewed as inferring latent variables with probabilistic distributions.

Summary

- Perception of high dimensional objects can be viewed as inferring latent variables with probabilistic distributions.
- We can impose structure.

Summary

- Perception of high dimensional objects can be viewed as inferring latent variables with probabilistic distributions.
- We can impose structure.
- We can learn through the inference process.
  - Taking the inference process into account.
  - Learning representations that matter.

Some Nuances

- If the process is deterministic or unimodal, standard deep networks may work.

Some Nuances

- If the process is deterministic or unimodal, standard deep networks may work.
- Network forward propagation vs. relaxed probabilistic inference.

Some Nuances

- If the process is deterministic or unimodal, standard deep networks may work.
- Network forward propagation vs. relaxed probabilistic inference.
- Having a stronger prior has the potential to be more data efficient.

Some Nuances

- If the process is deterministic or unimodal, standard deep networks may work.
- Network forward propagation vs. relaxed probabilistic inference.
- Having a stronger prior has the potential to be more data efficient.
- And you will need structured / generative learning when there are multiple modes.
  - E.g. Planning: there can be multiple future trajectories

Autoregressive Modeling

- Another type of output is autoregressive modeling.

Autoregressive Modeling

- Another type of output is autoregressive modeling.
- Example: Object detection/segmentation.

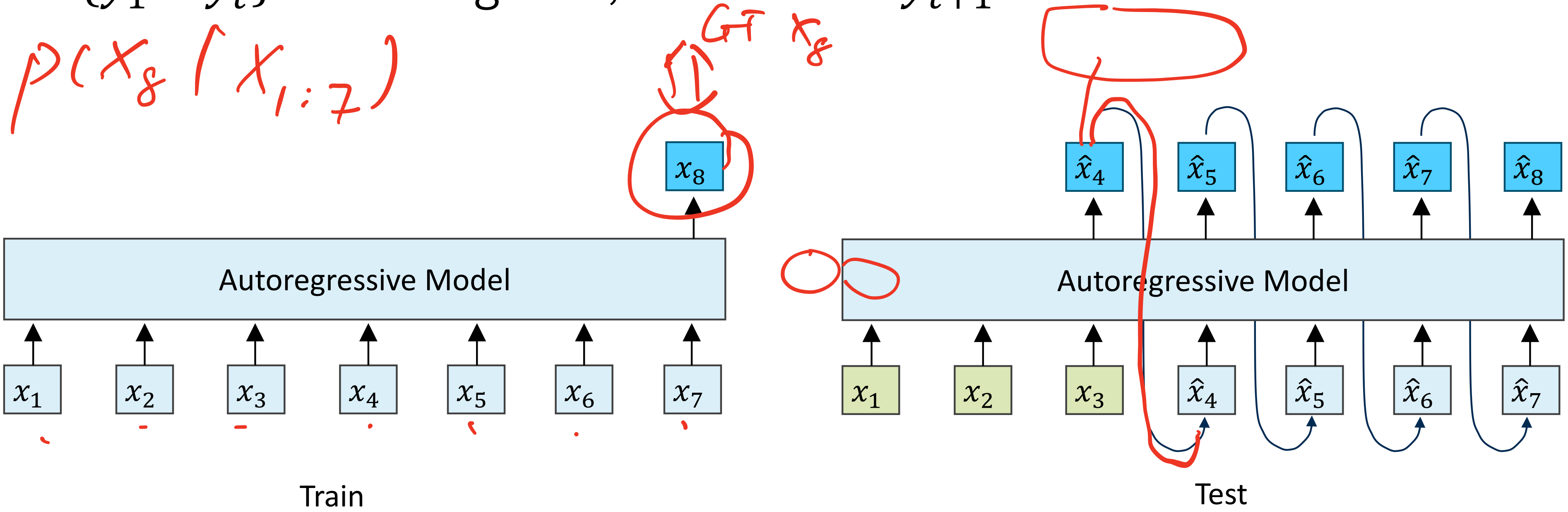
Autoregressive Modeling

- Another type of output is autoregressive modeling.
- Example: Object detection/segmentation.
- Intuition: Our visual attention focus on one object at a time.

Teacher Forcing

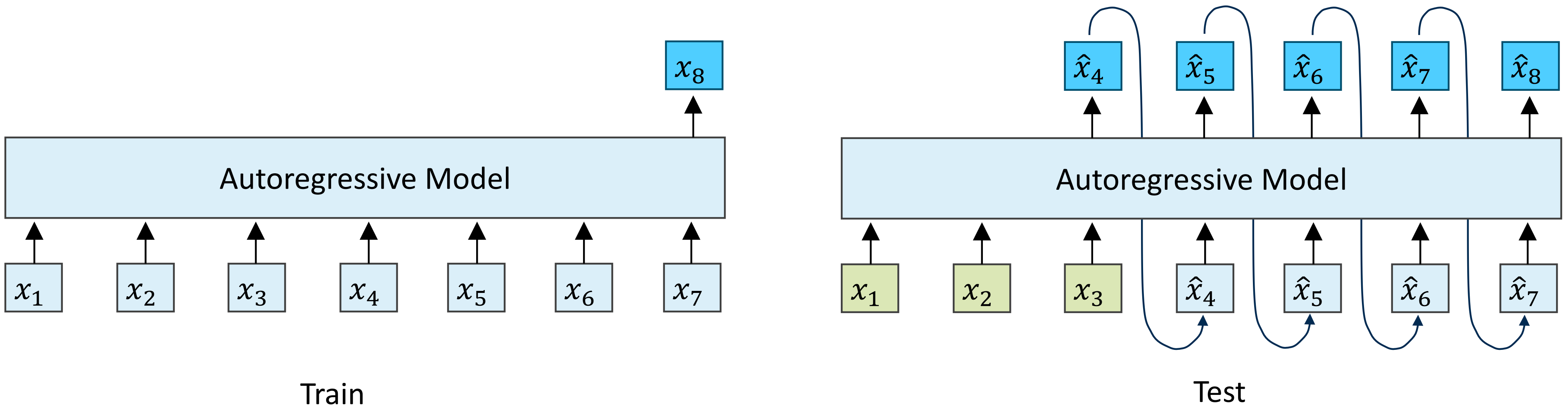
- Teacher Forcing: Pretend that you know the whole sequence $\{y_1 \dots y_t\}$ at training time, and train for y_{t+1}.

$p(x_8 \mid x_{1:7})$



Teacher Forcing

- Teacher Forcing: Pretend that you know the whole sequence $\{y_1 \ldots y_t\}$ at training time, and train for y_{t+1}.
- Problem: You have to know the ordering.



More on Ordering

- There are many set to set problems.

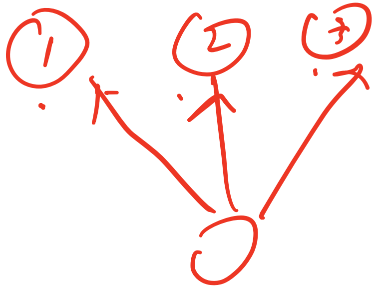
More on Ordering

C

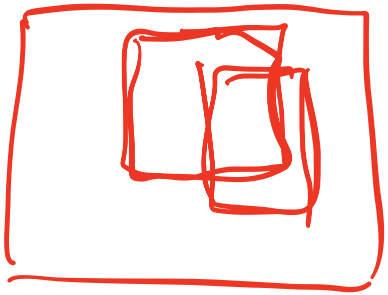
A B

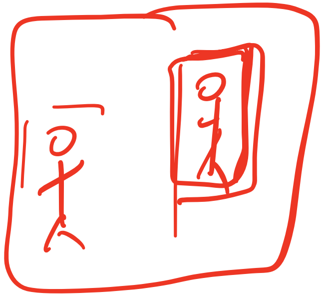
- There are many set to set problems.
- E.g. Detection, segmentation, generating multiple objects, clustering

More on Ordering



- There are many set to set problems.
- E.g. Detection, segmentation, generating multiple objects, clustering
- Input does not follow a particular order. Loss function should not favor a particular order.
  - Attention: The attention operation is order-invariant.
  - Matching: The teacher can match the next input based on the closest matching groundtruth instances.

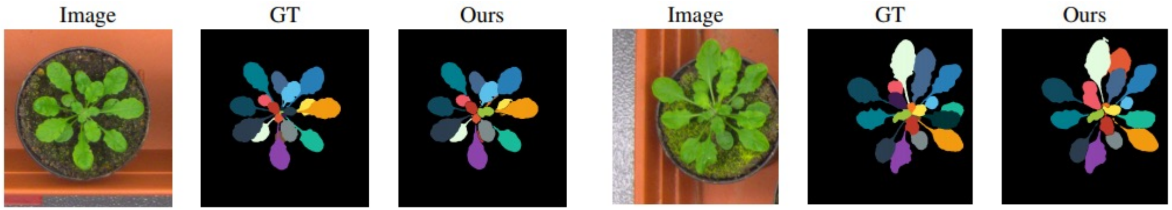


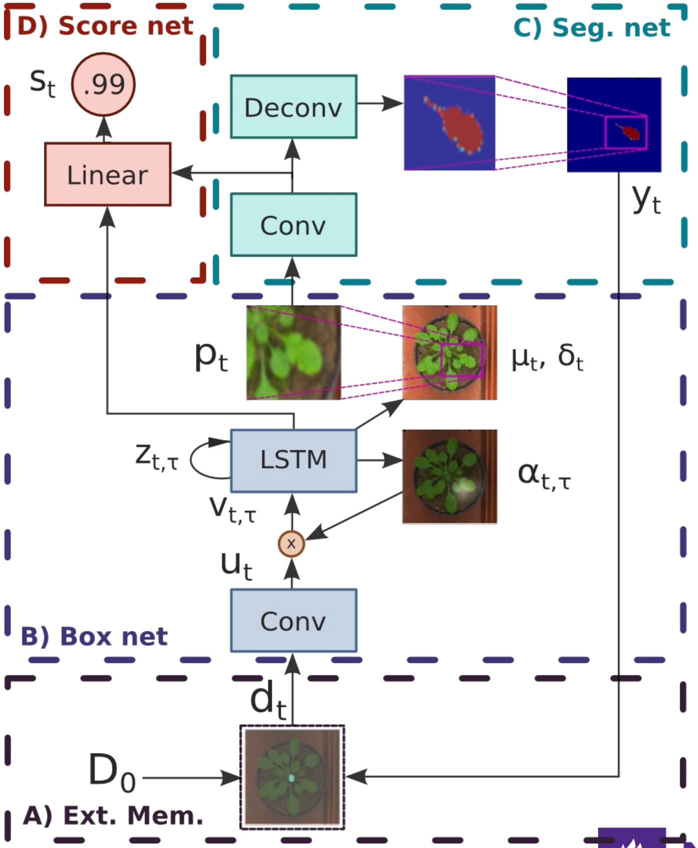


NYU

Instance Segmentation using Attention

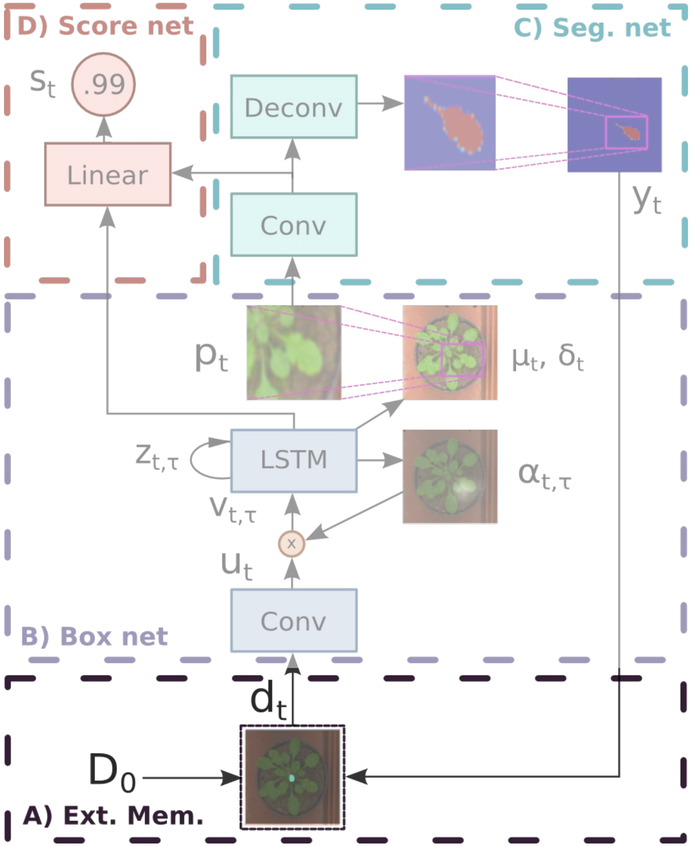
- Attending to one region at a time.
- Zooming in for segmentation.
- End-to-end differentiable box proposals and external memory.
- State-of-the-art on leave segmentation problems for many years.





NYU

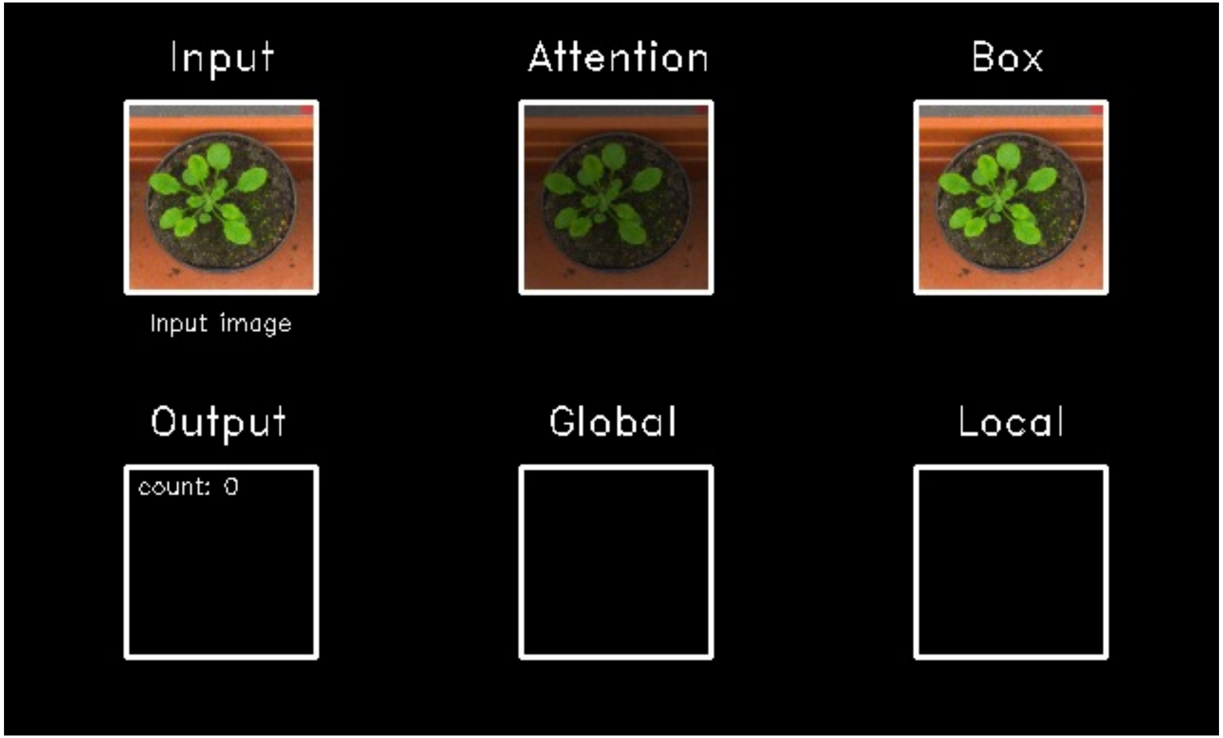
Demo

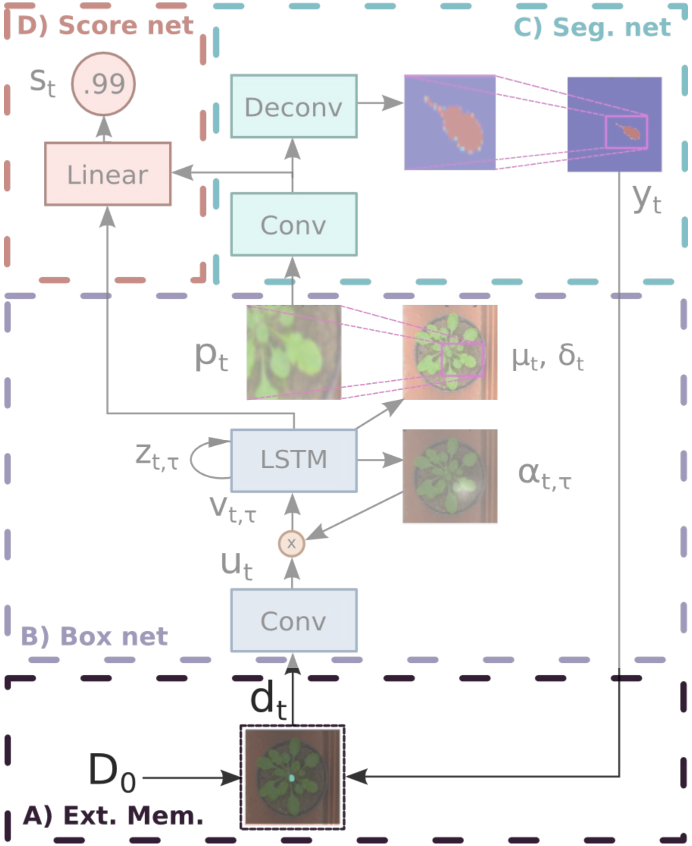


[Ren & Zemel, 2017]

NYU

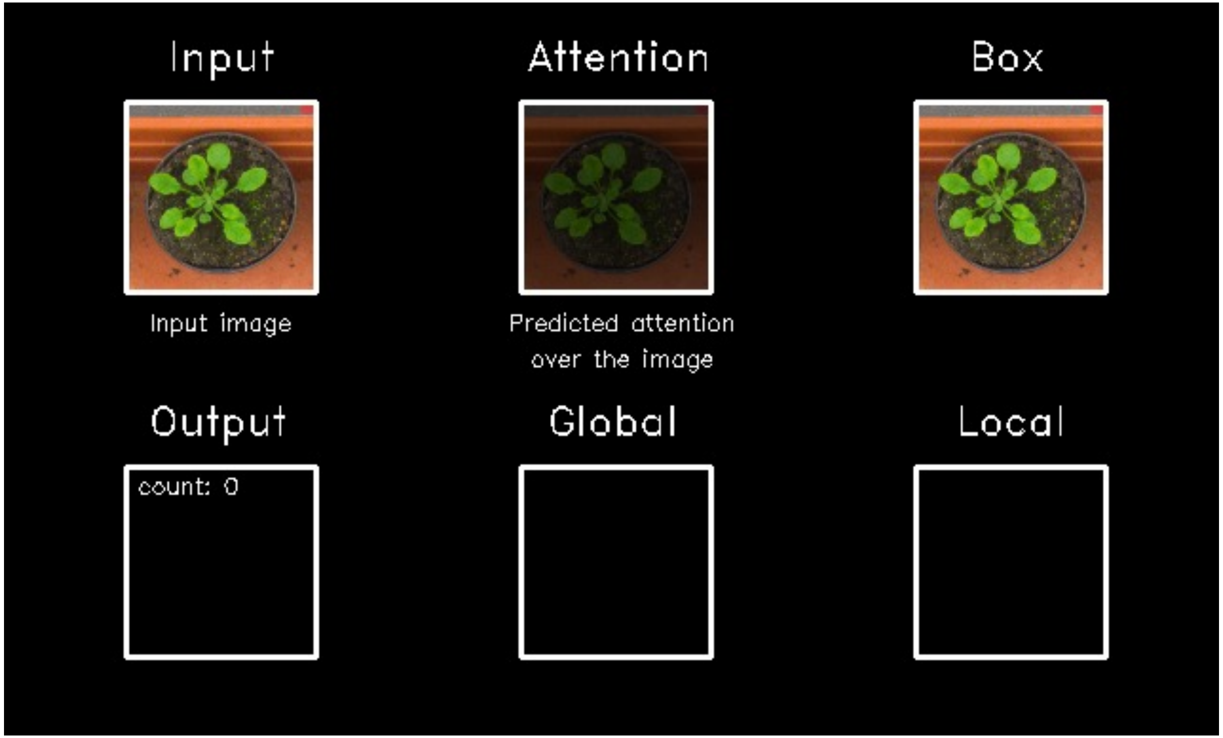
Demo

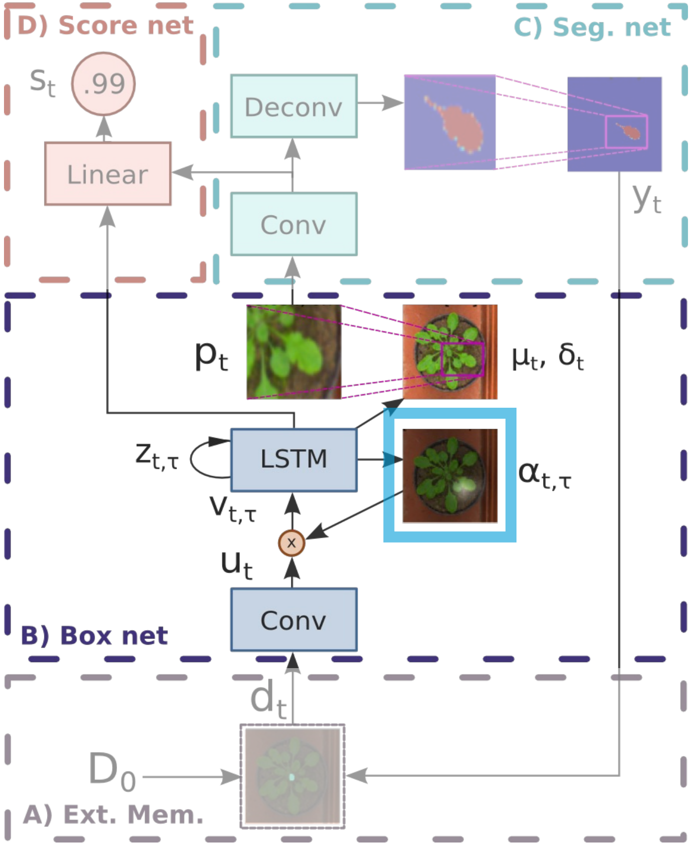




[Ren & Zemel, 2017]

Demo

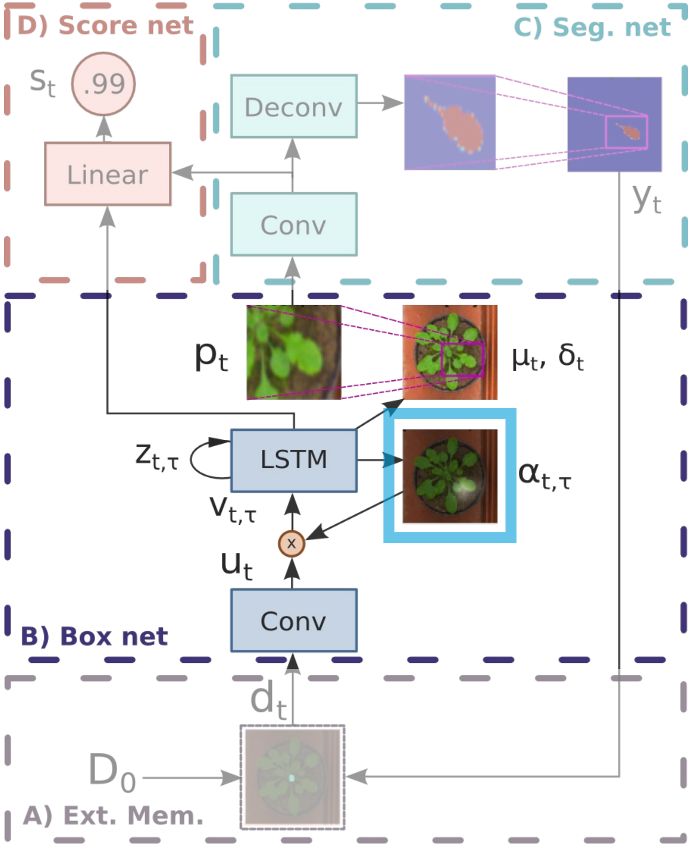




[Ren & Zemel, 2017]

NYU

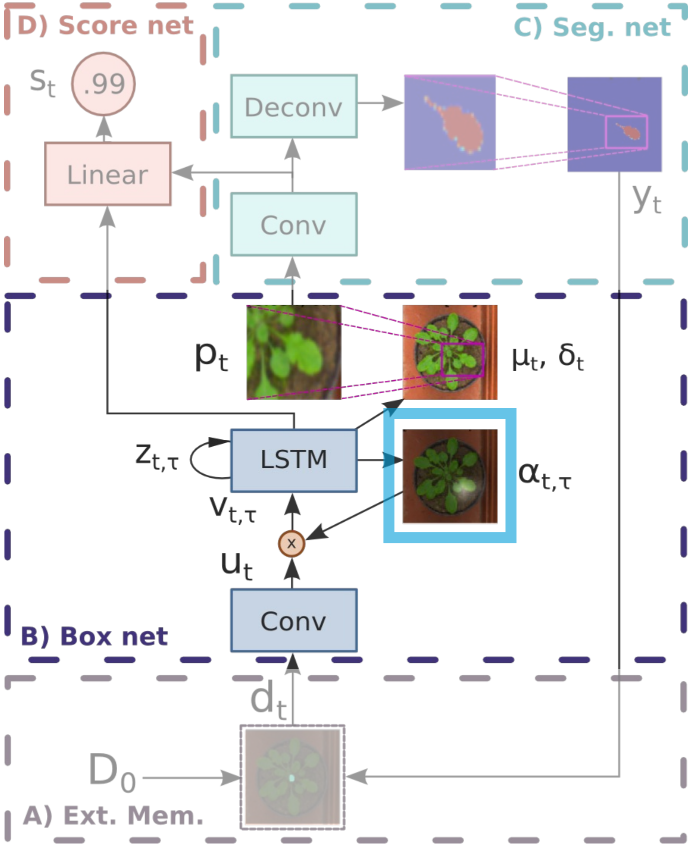
Demo



[Ren & Zemel, 2017]

NYU

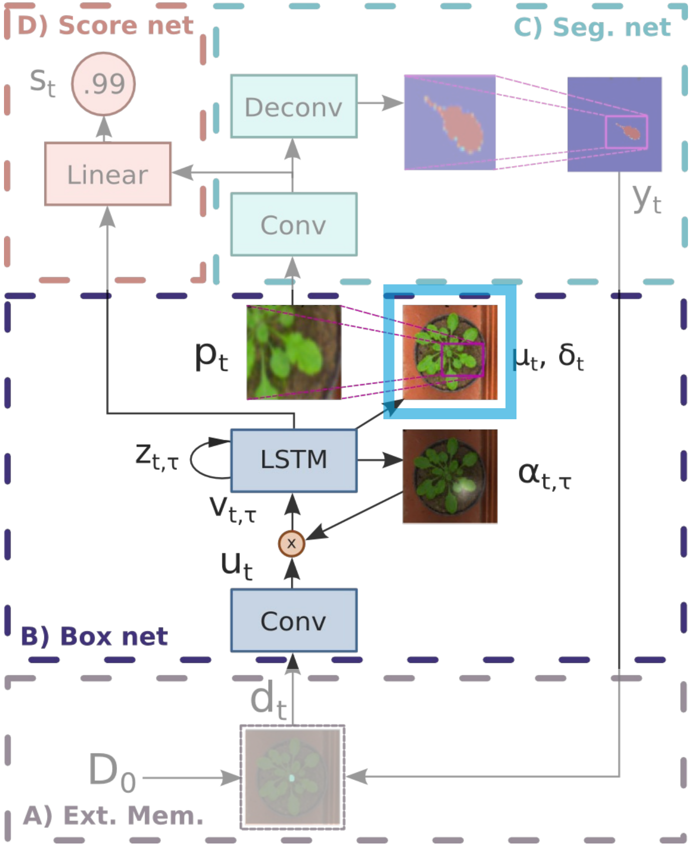
Demo



[Ren & Zemel, 2017]

NYU

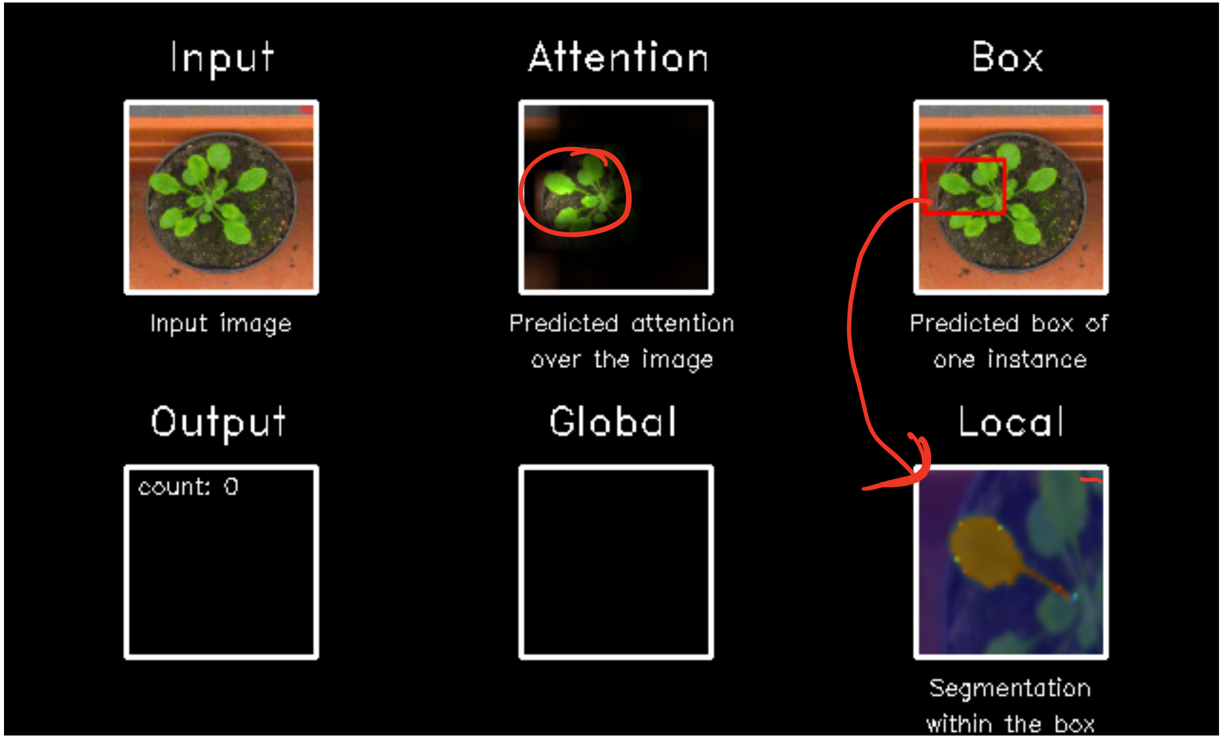
Demo

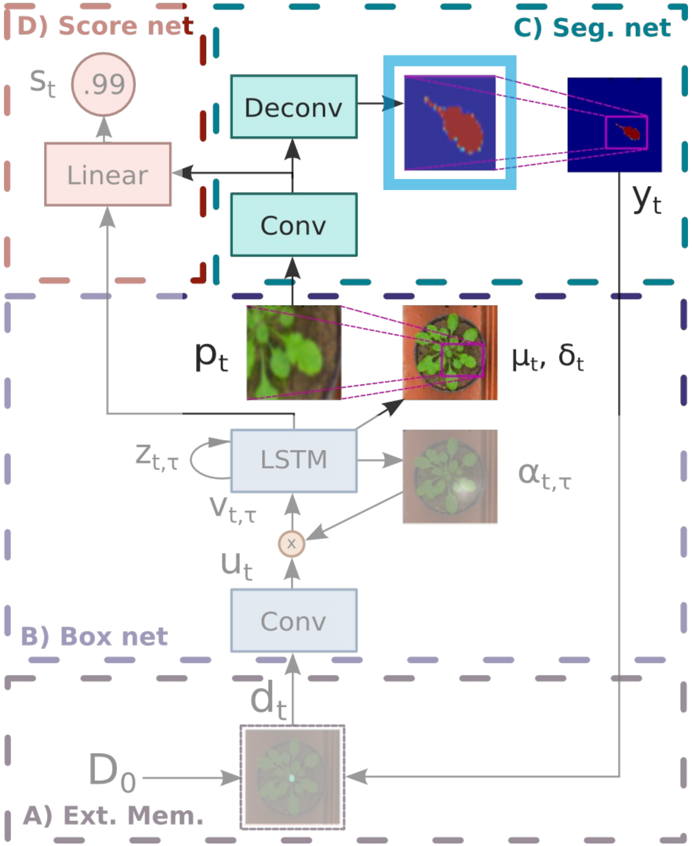


[Ren & Zemel, 2017]

NYU

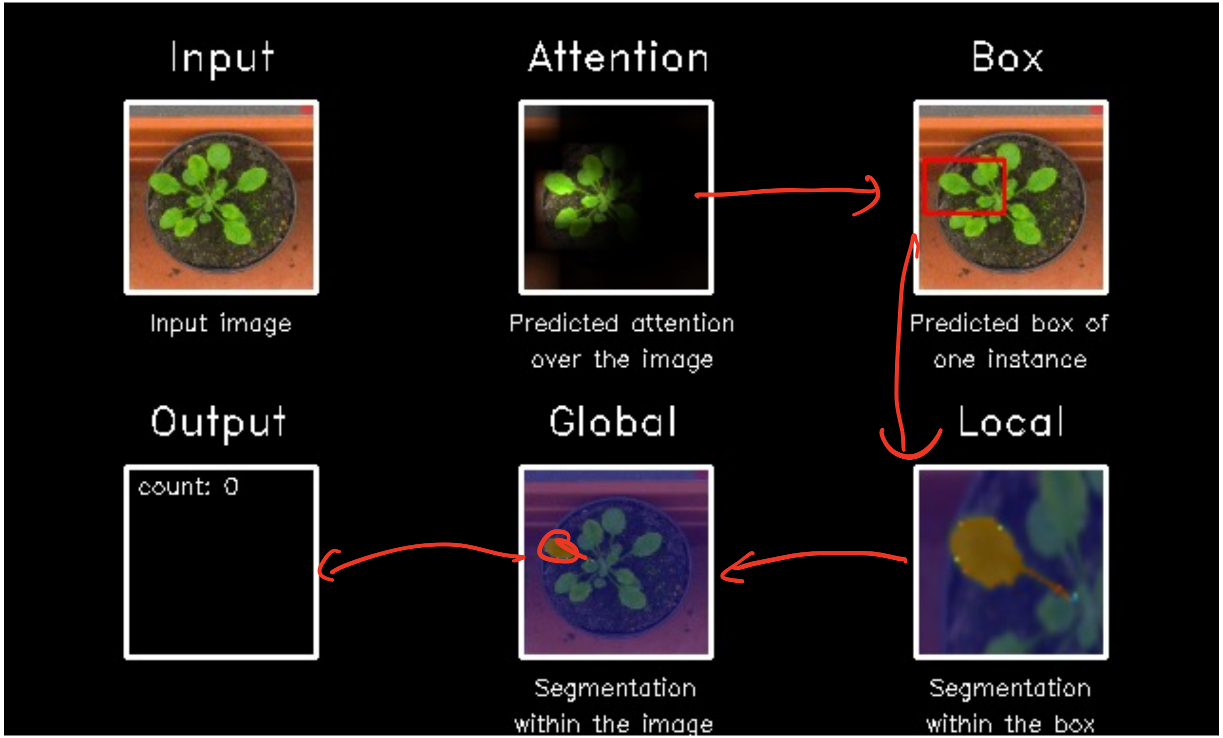
Demo

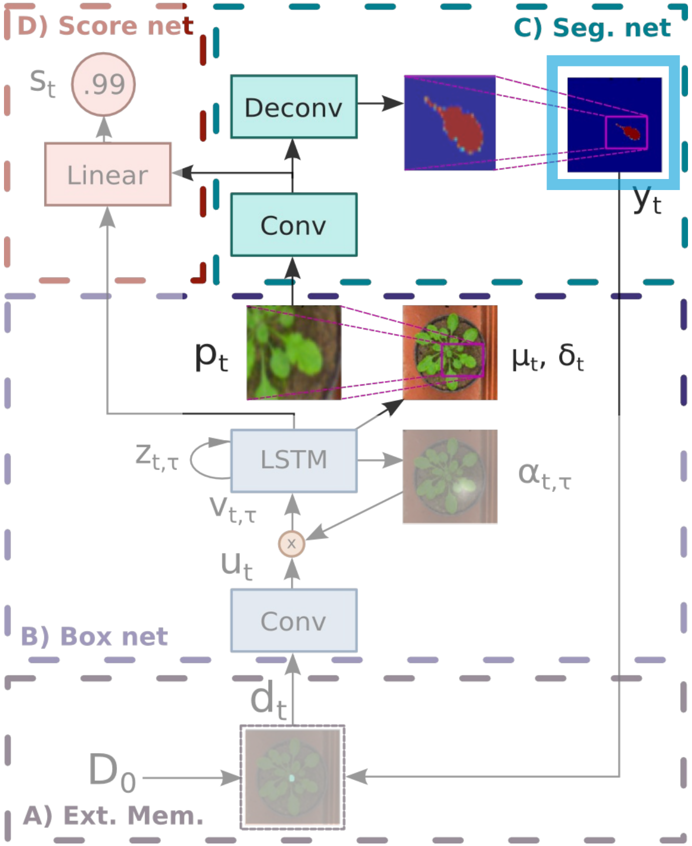




[Ren & Zemel, 2017]

Demo

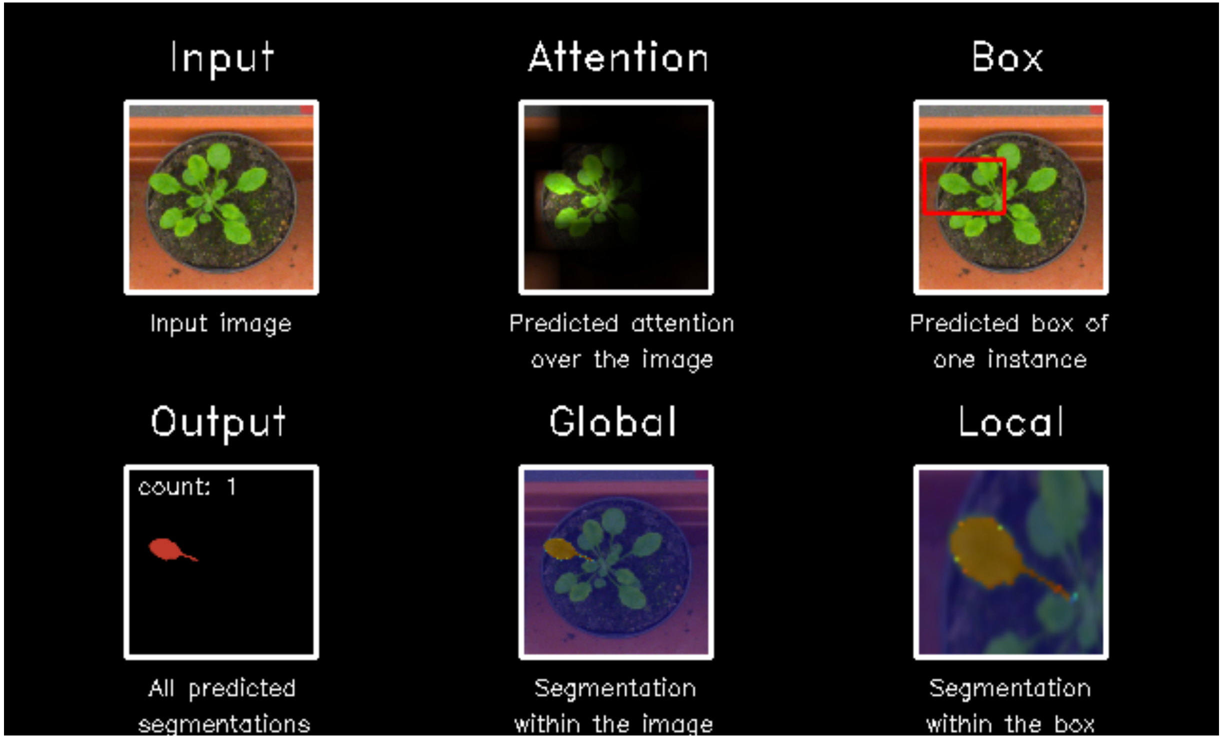


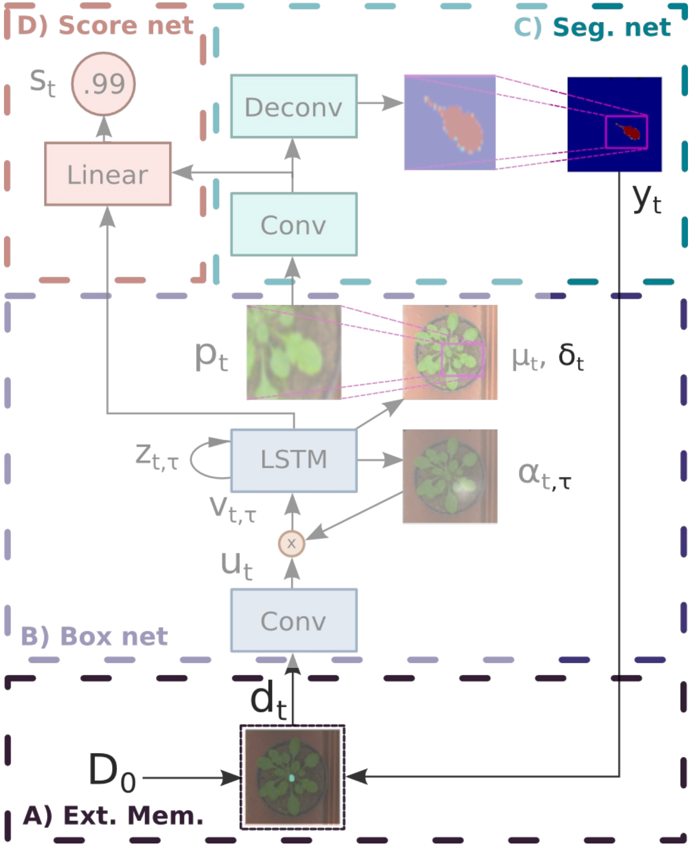


[Ren & Zemel, 2017]

NYU

Demo

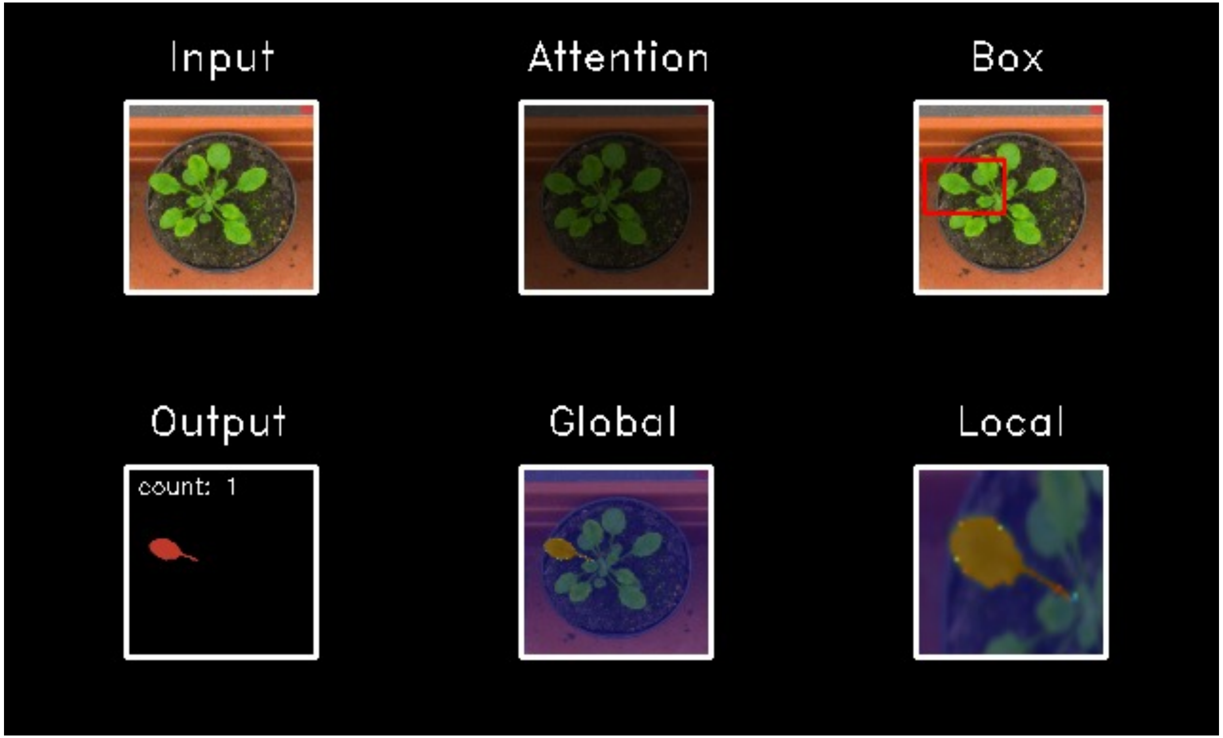


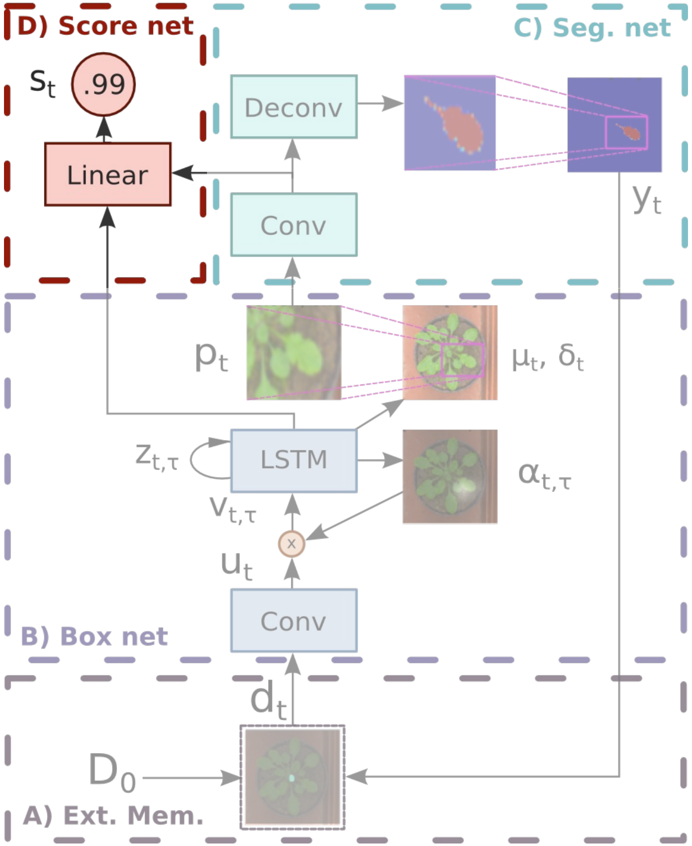


[Ren & Zemel, 2017]

NYU

Demo

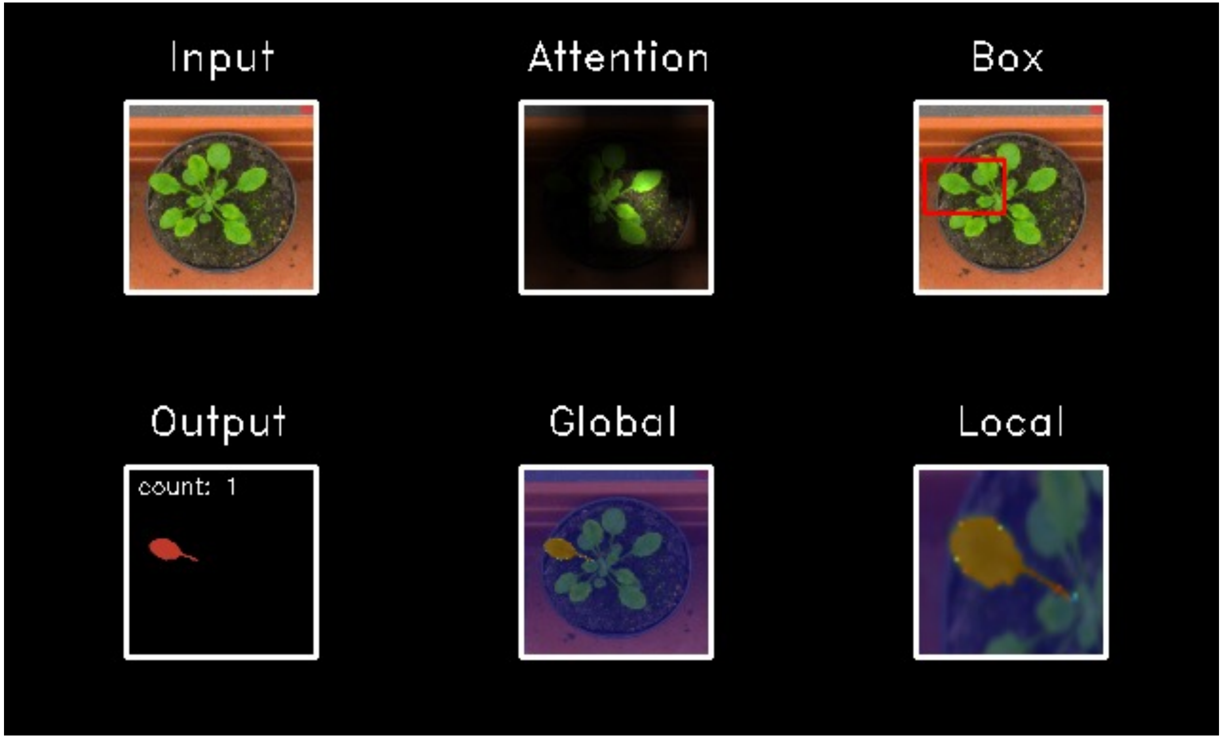


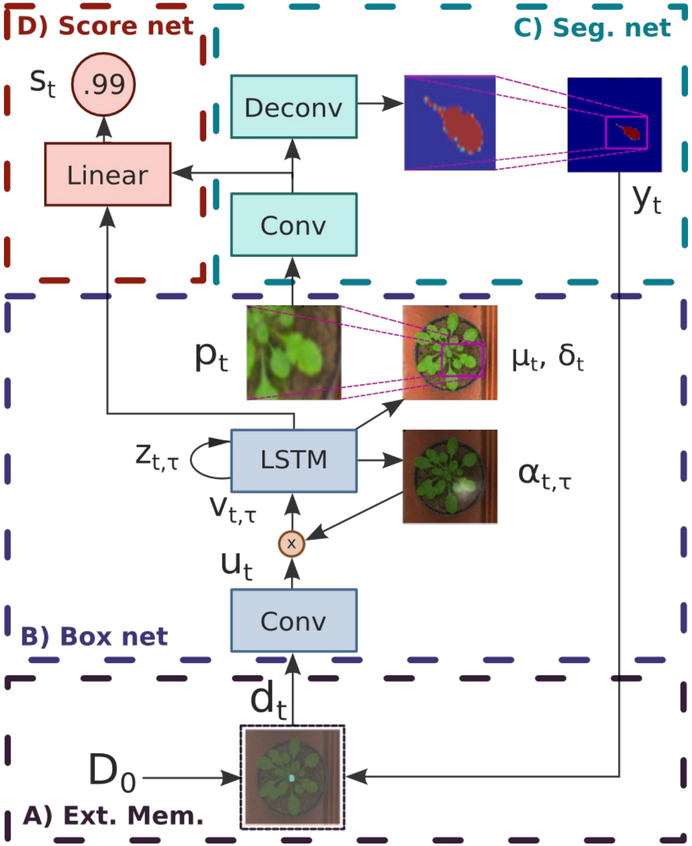


[Ren & Zemel, 2017]

NYU

Demo

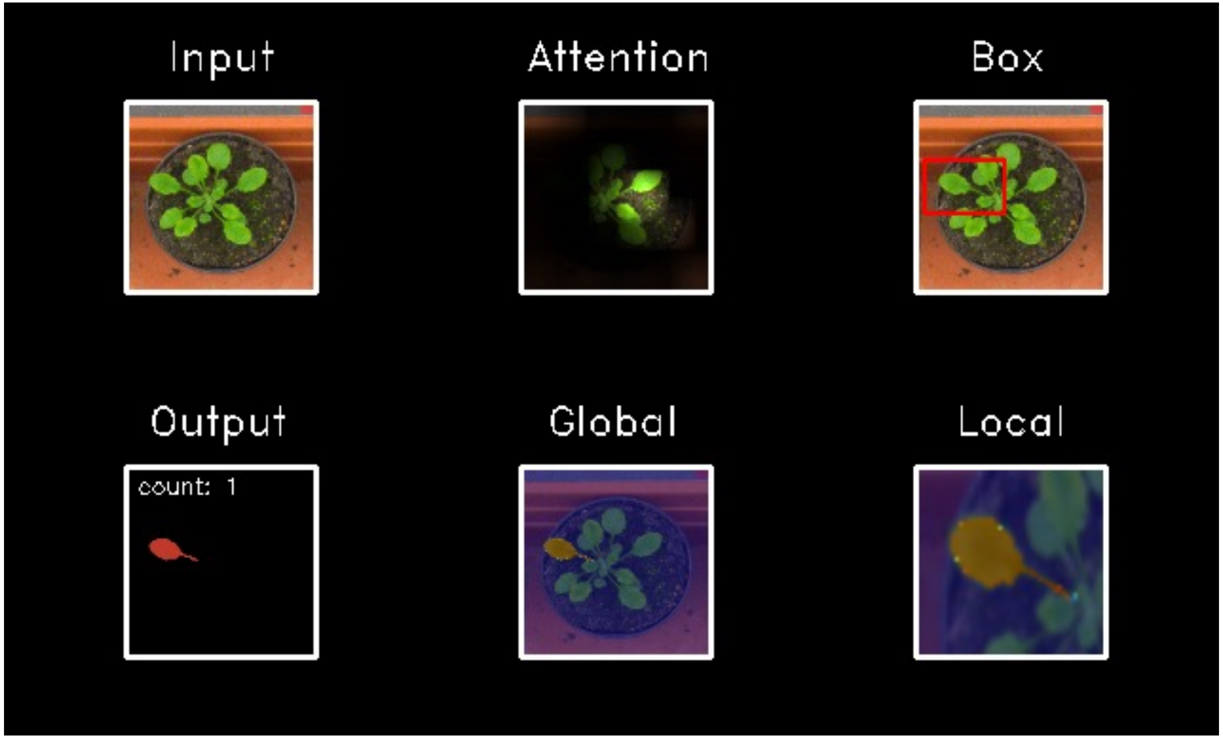


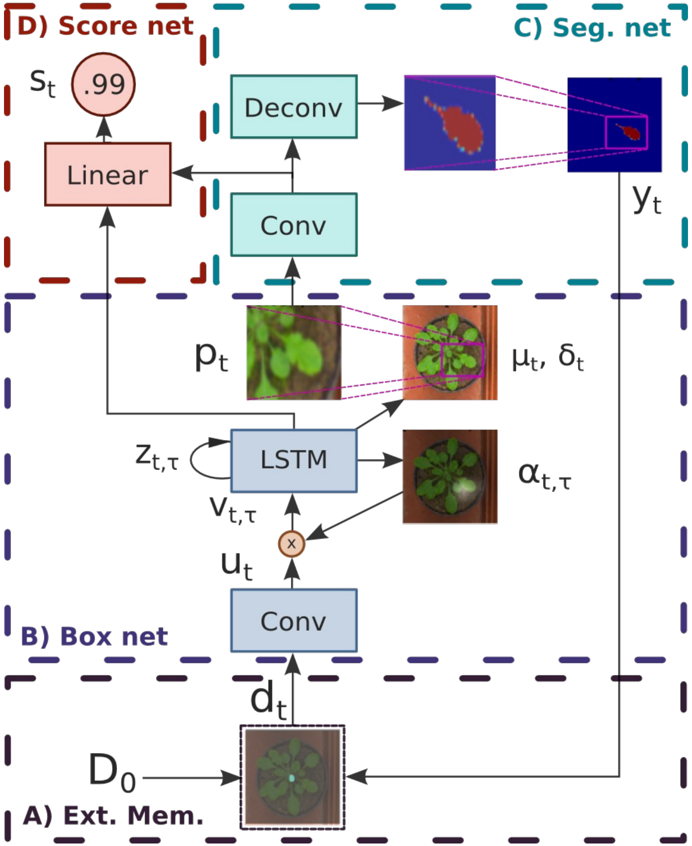


[Ren & Zemel, 2017]

NYU

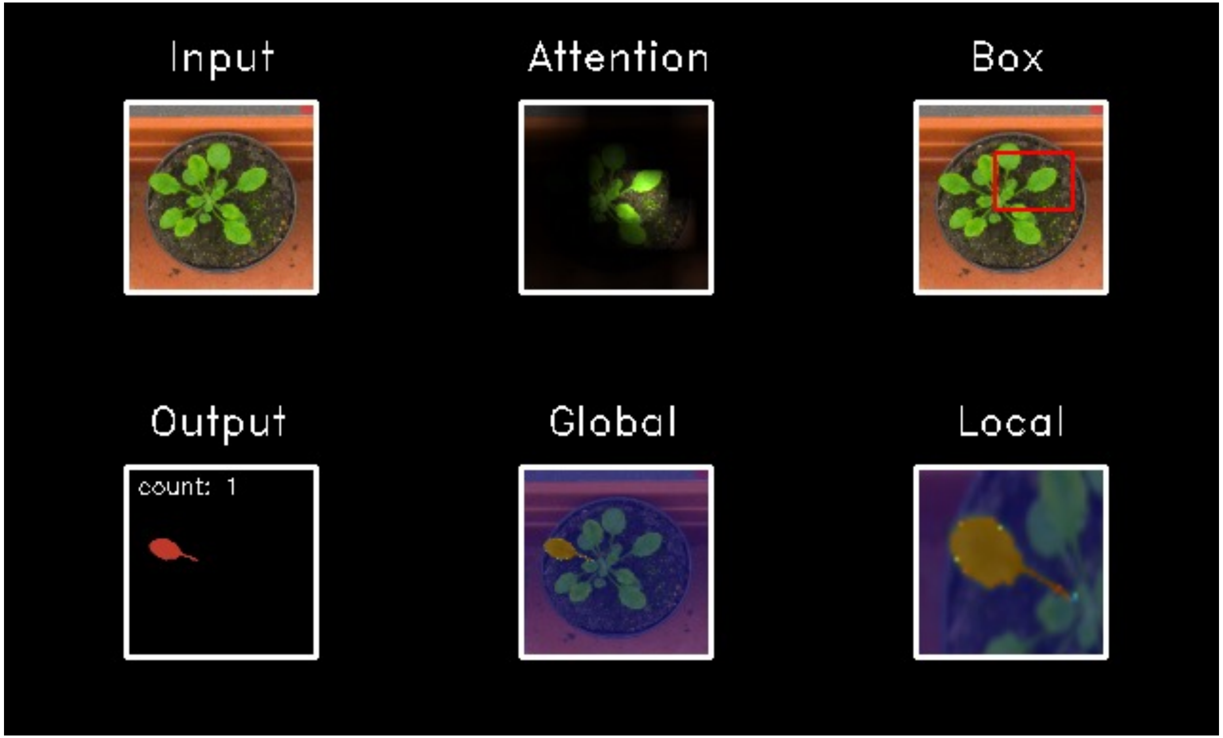
Demo

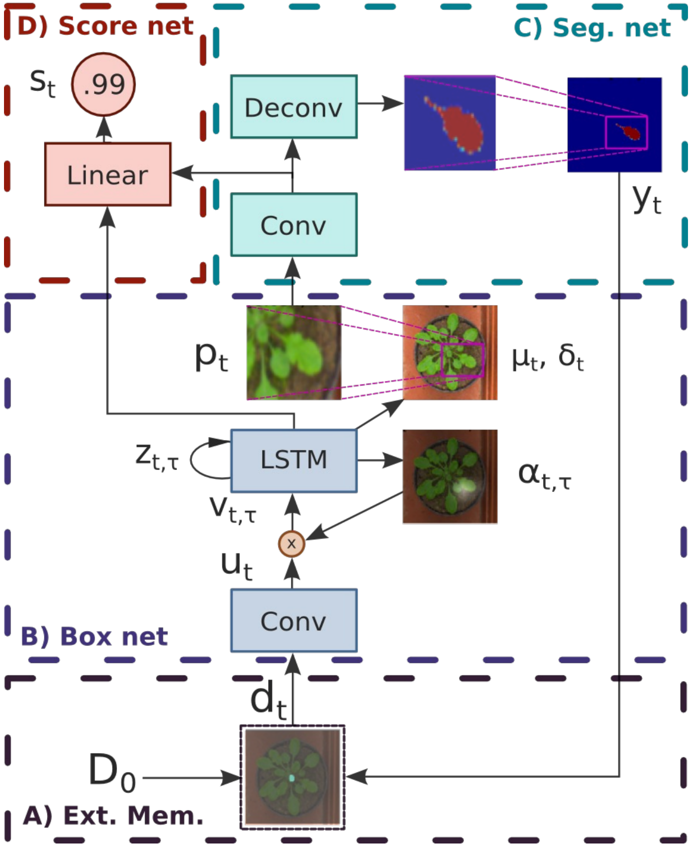




[Ren & Zemel, 2017]

Demo

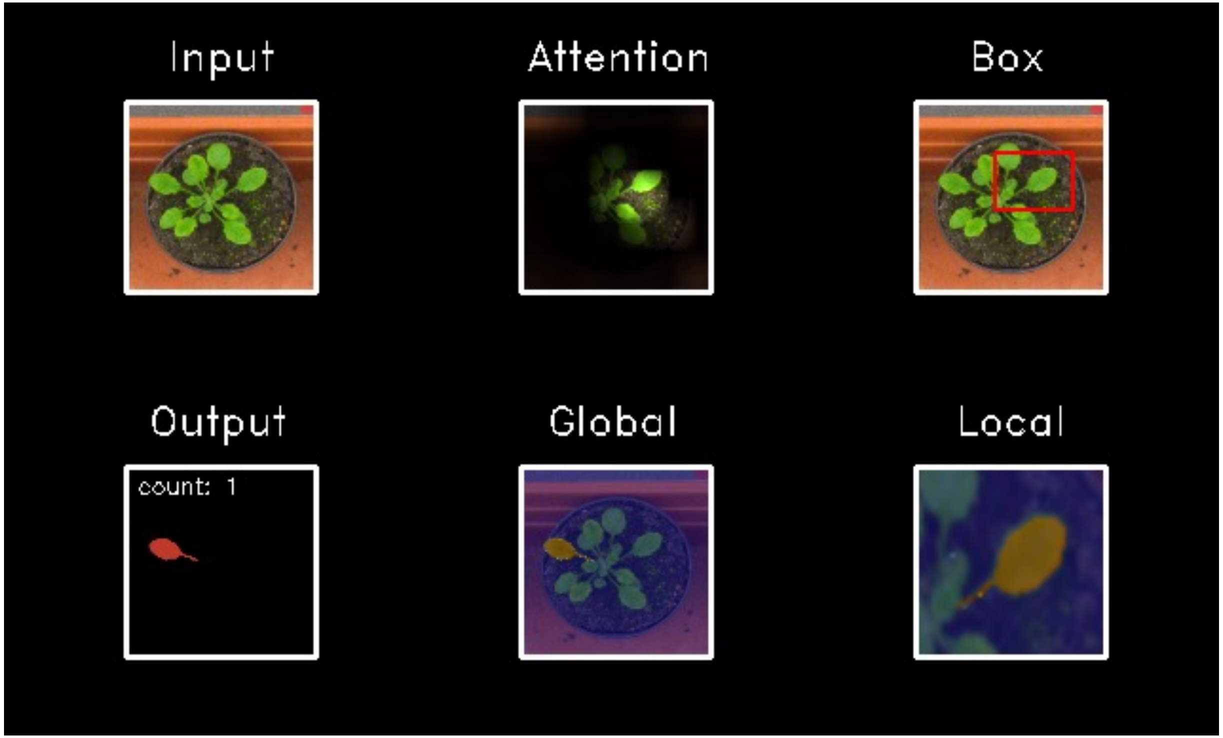


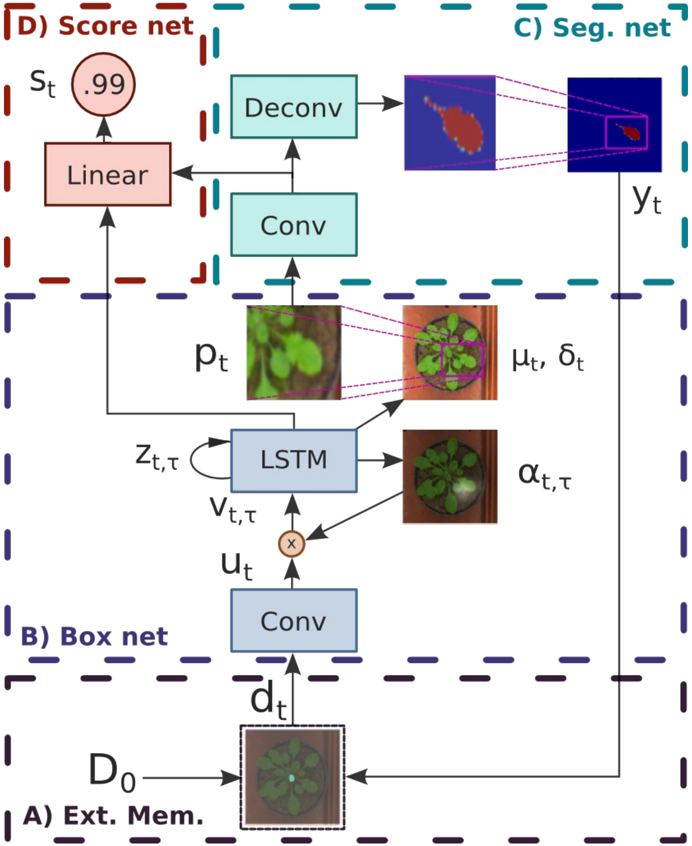


[Ren & Zemel, 2017]

NYU

Demo

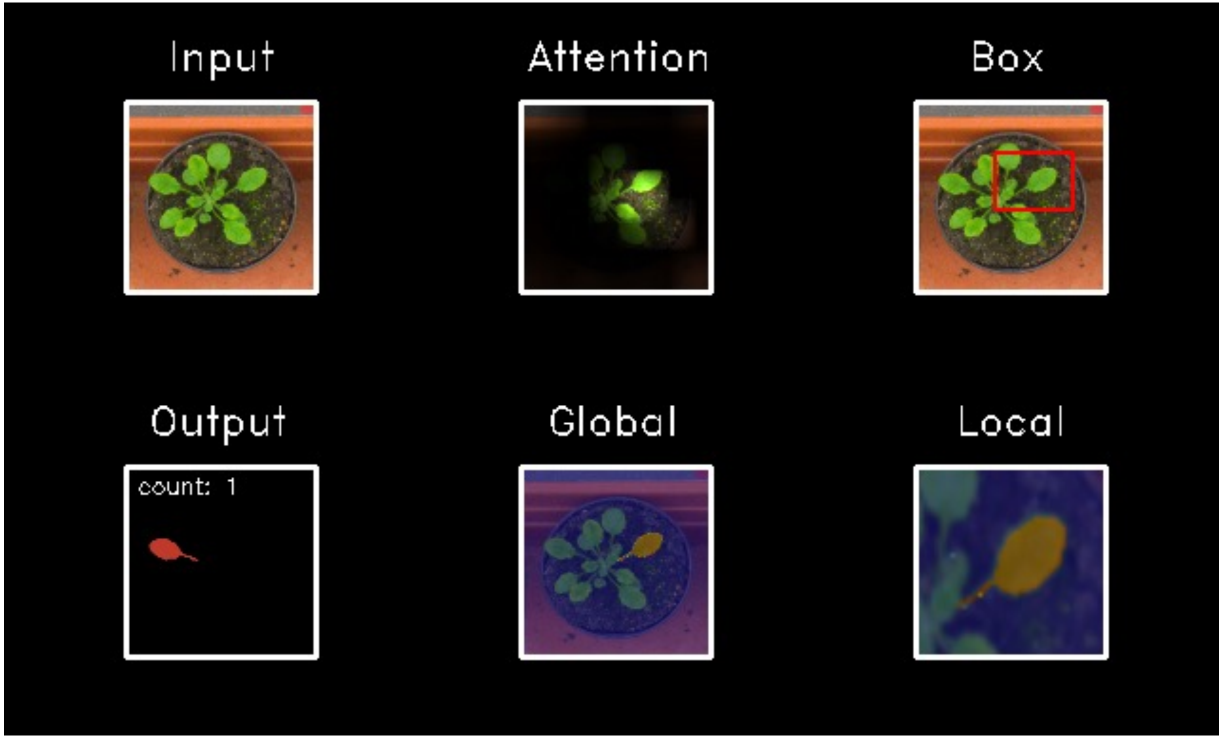


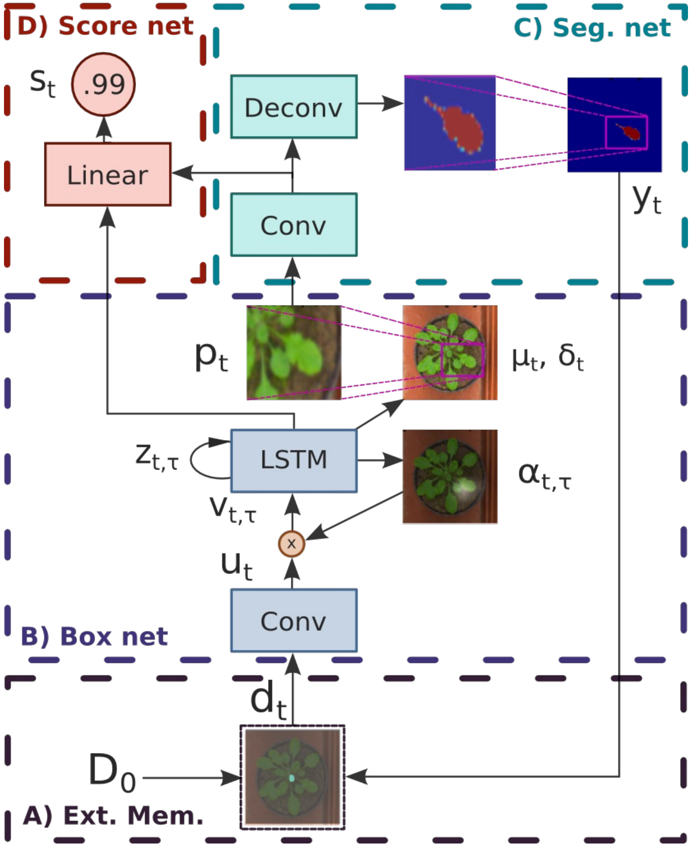


[Ren & Zemel, 2017]

NYU

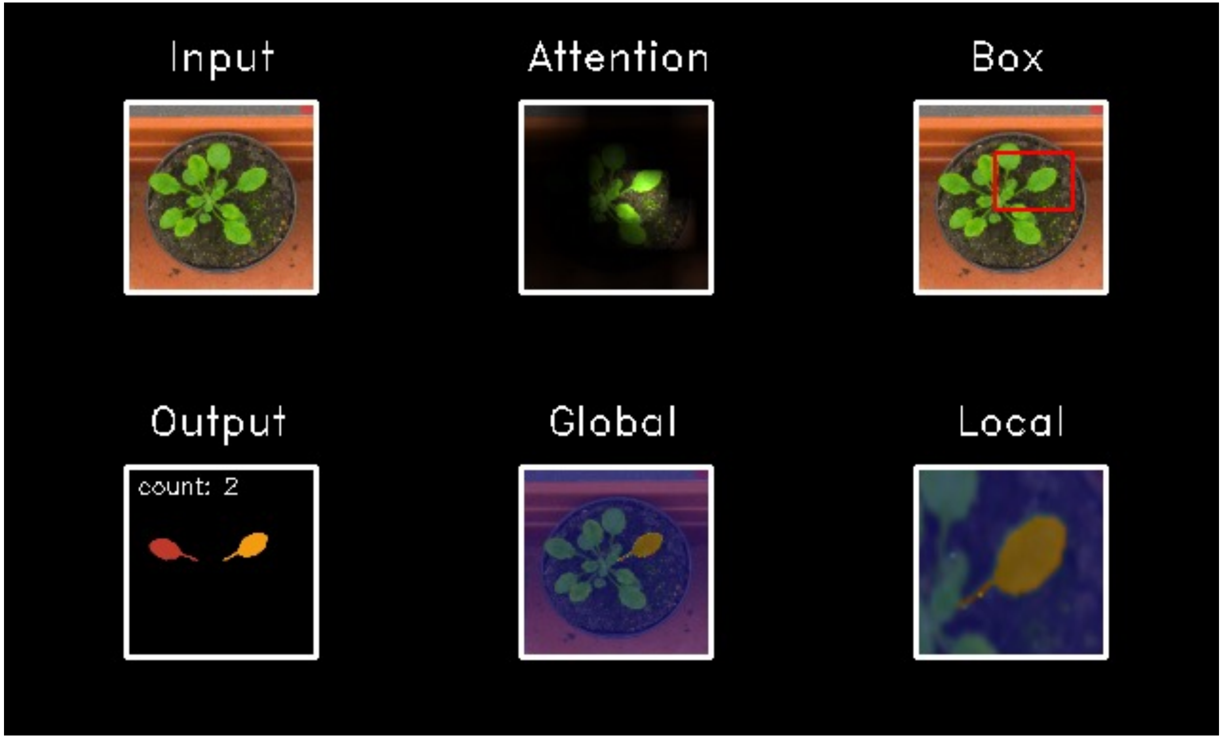
Demo

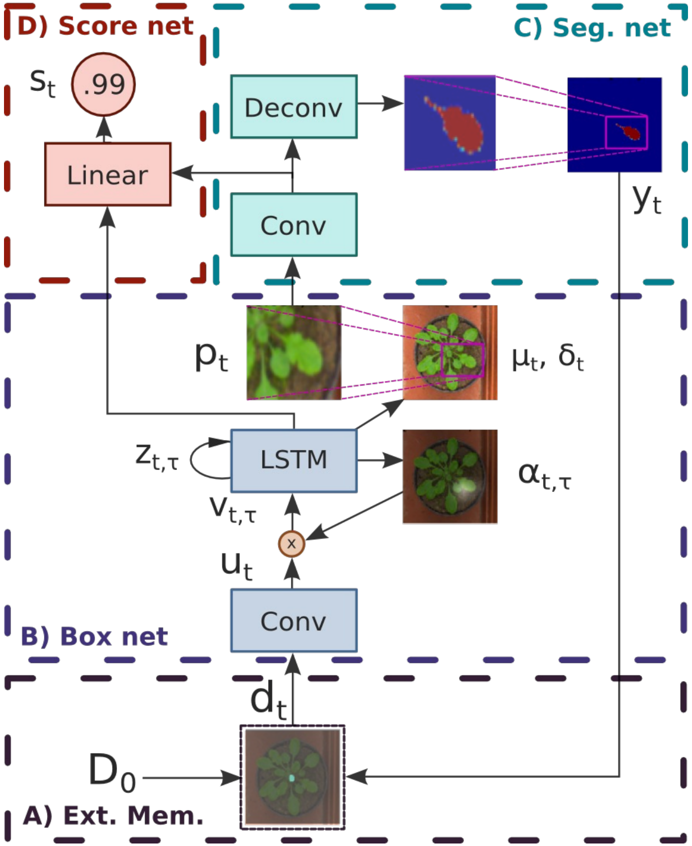




[Ren & Zemel, 2017]

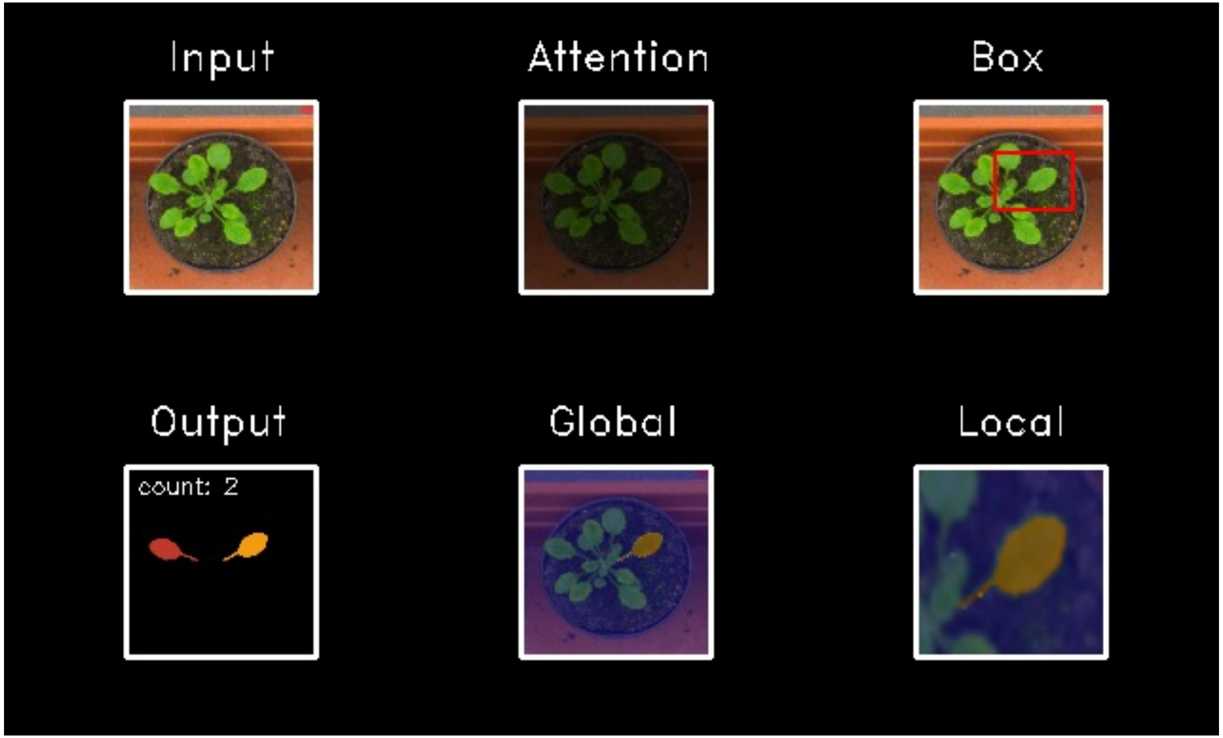
Demo

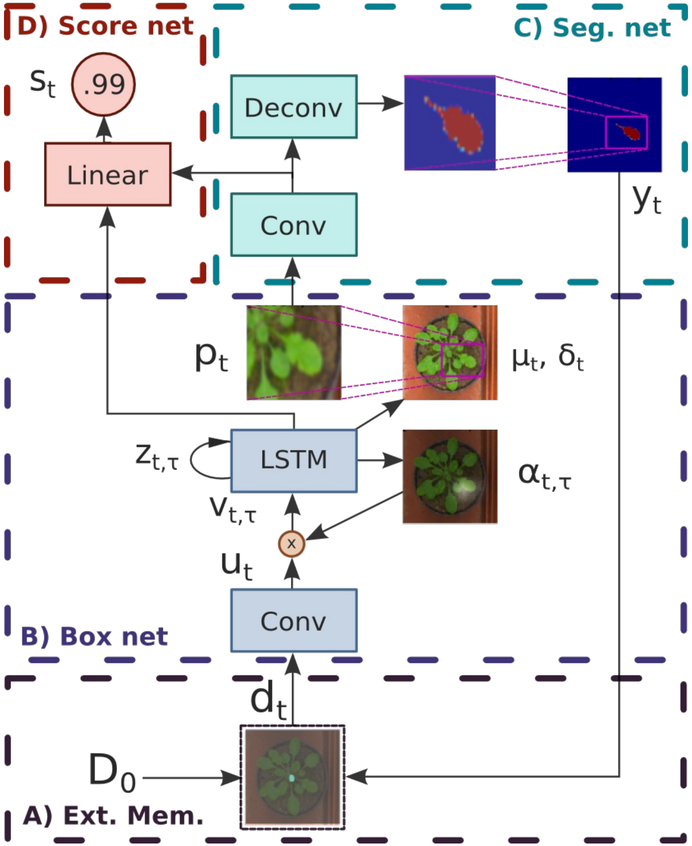




[Ren & Zemel, 2017]

Demo

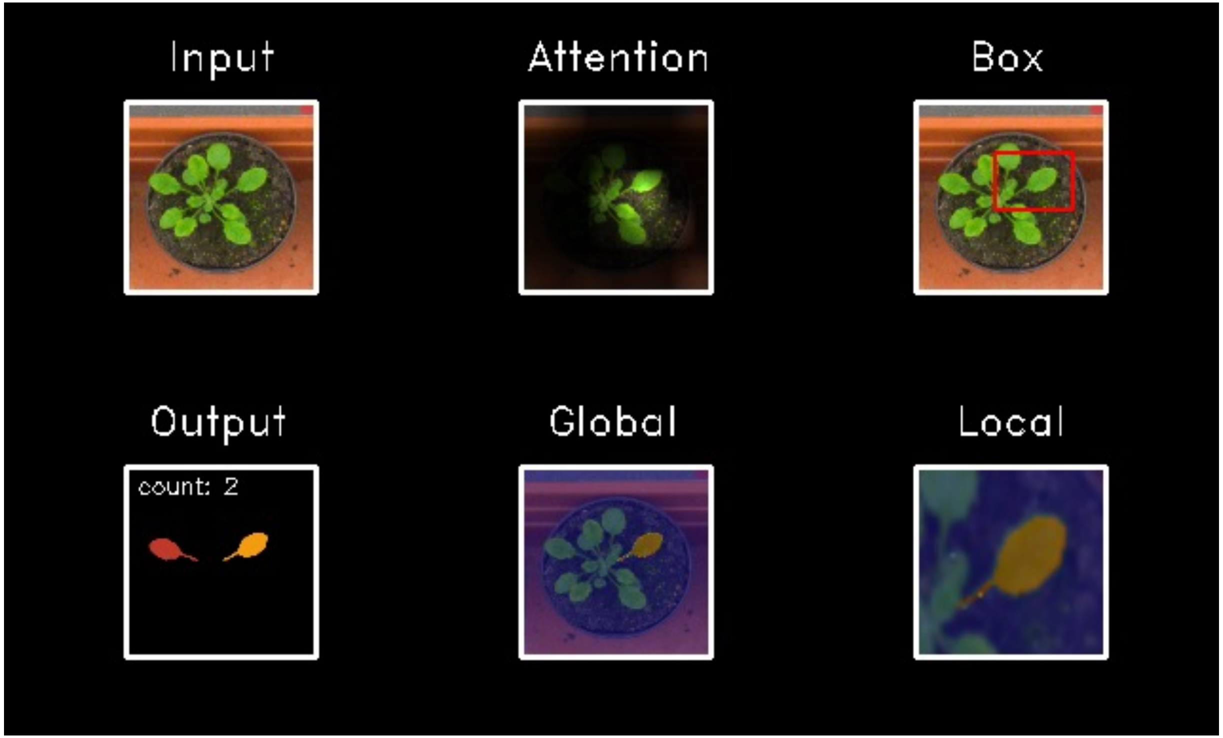


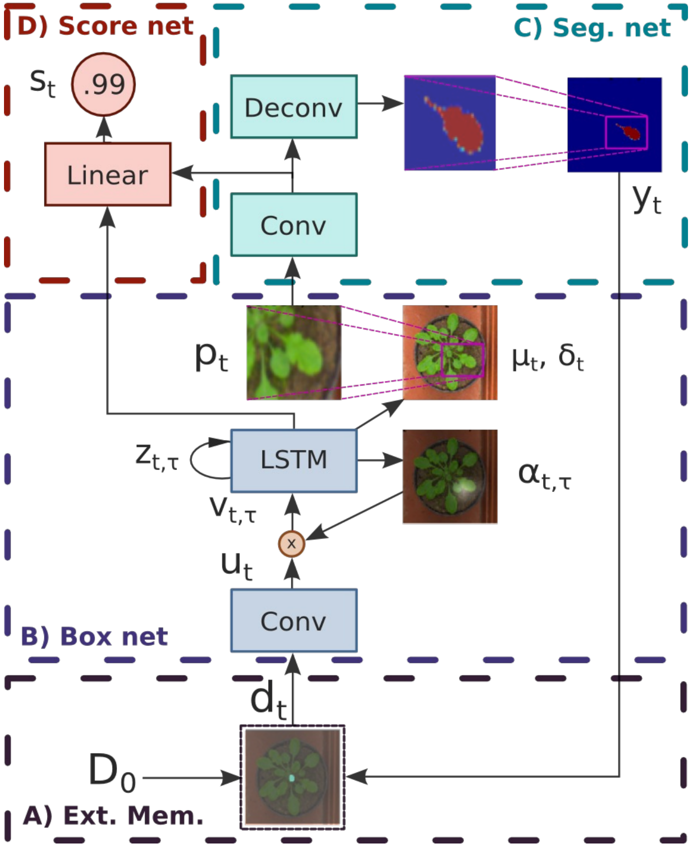


[Ren & Zemel, 2017]

NYU

Demo

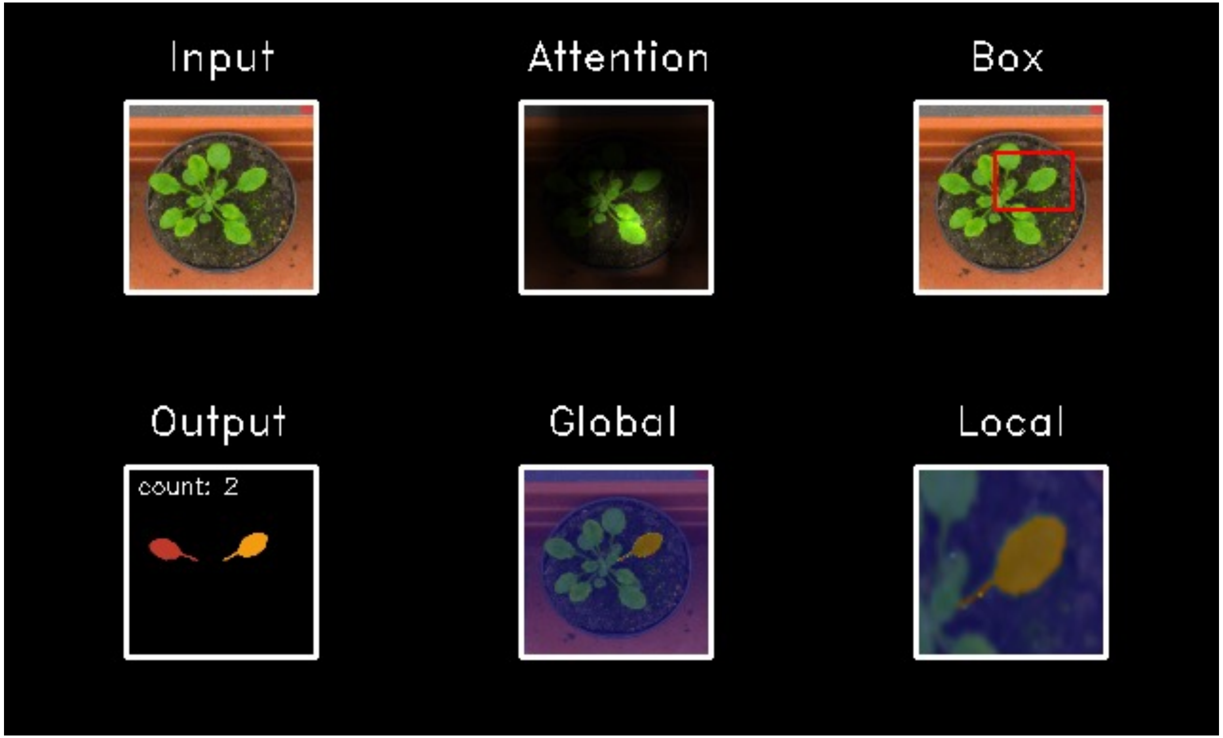


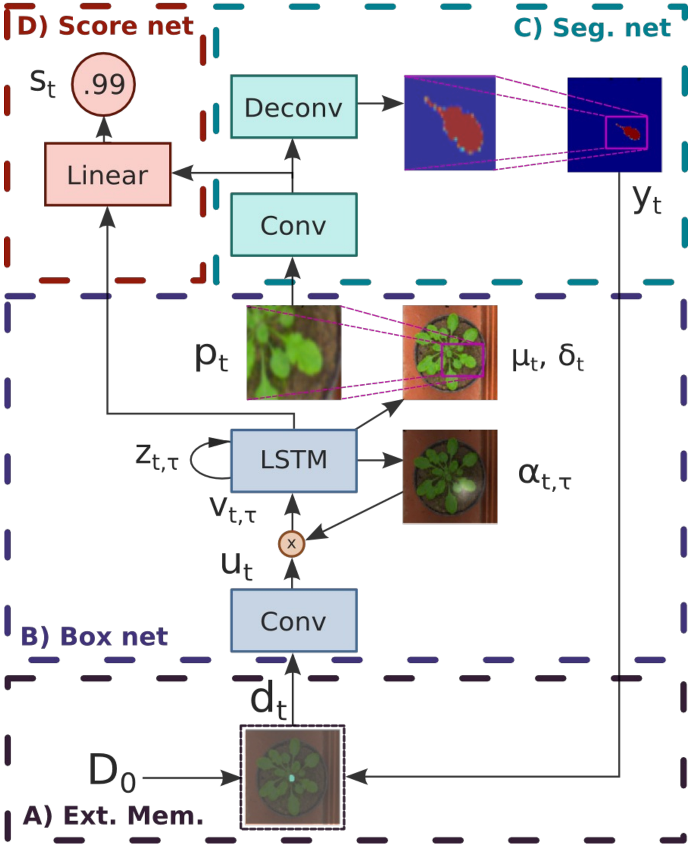


[Ren & Zemel, 2017]

NYU

Demo

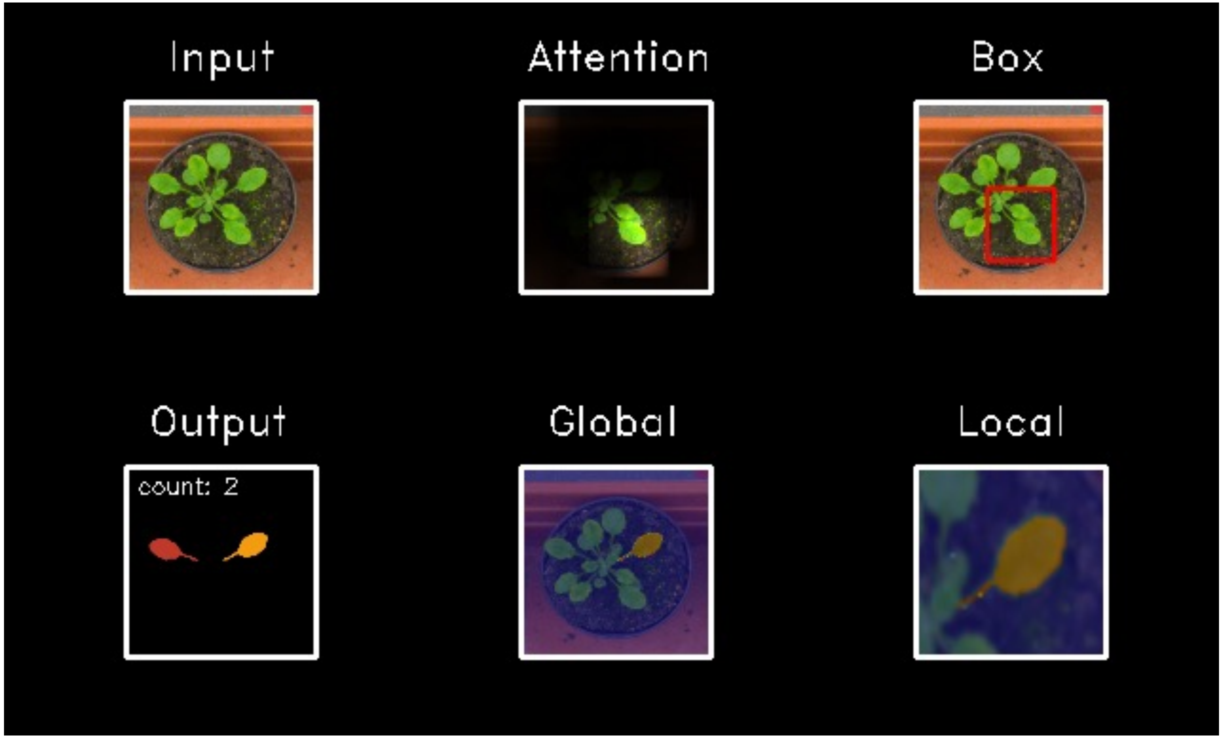


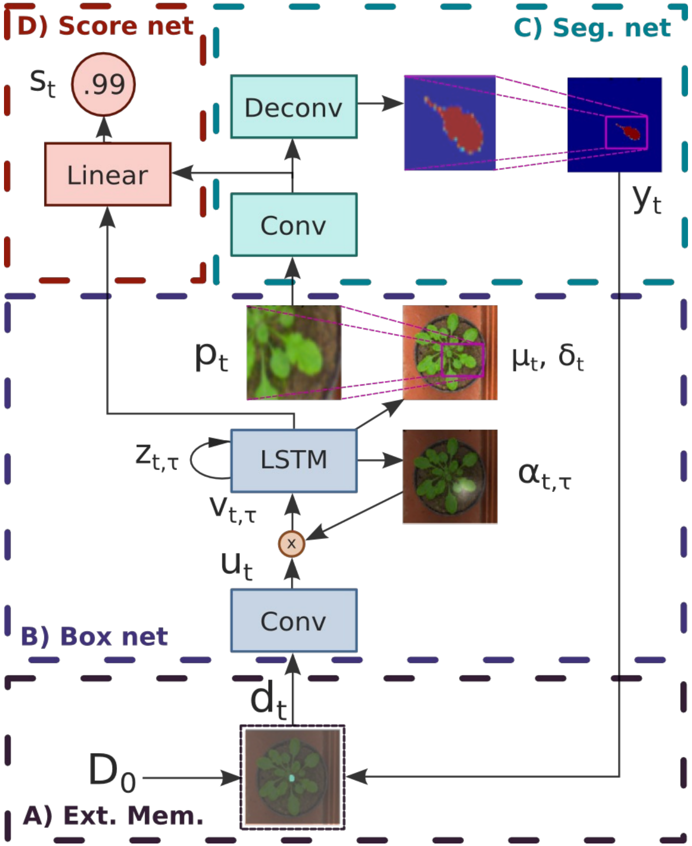


[Ren & Zemel, 2017]

NYU

Demo

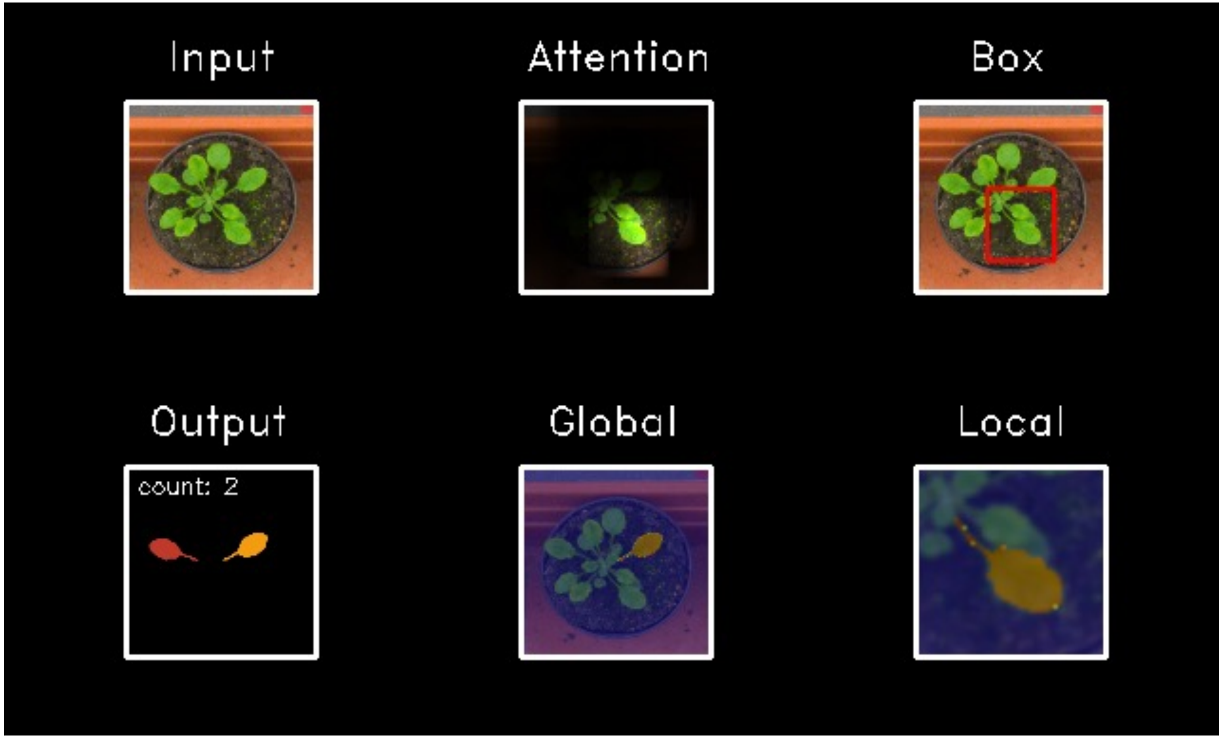


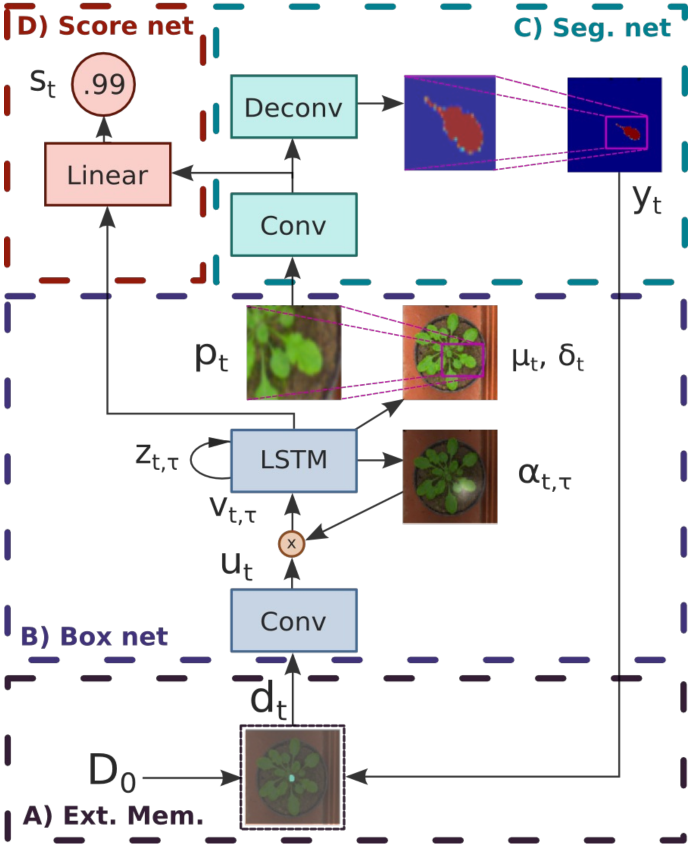


[Ren & Zemel, 2017]

NYU

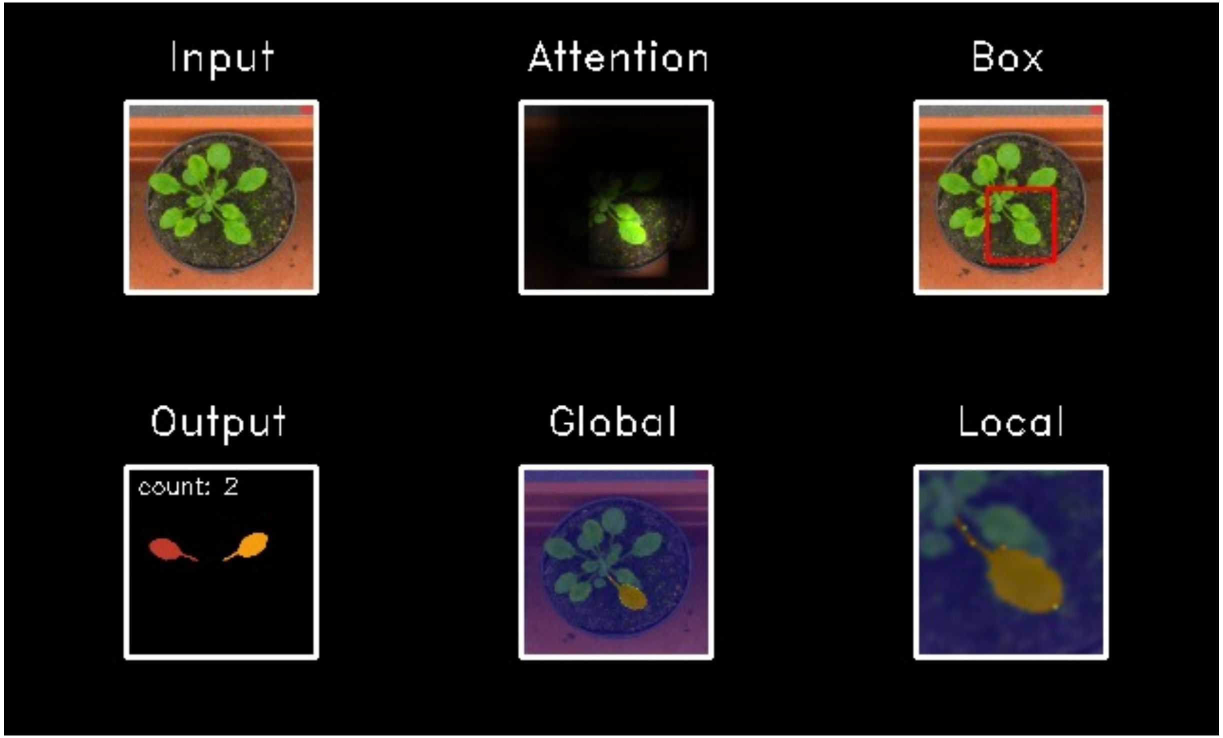
Demo

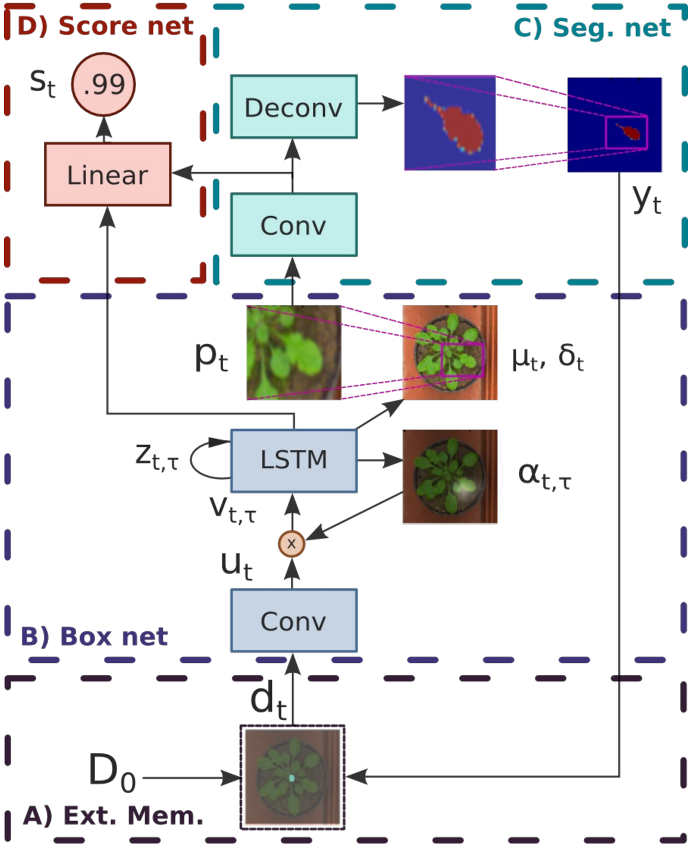




[Ren & Zemel, 2017]

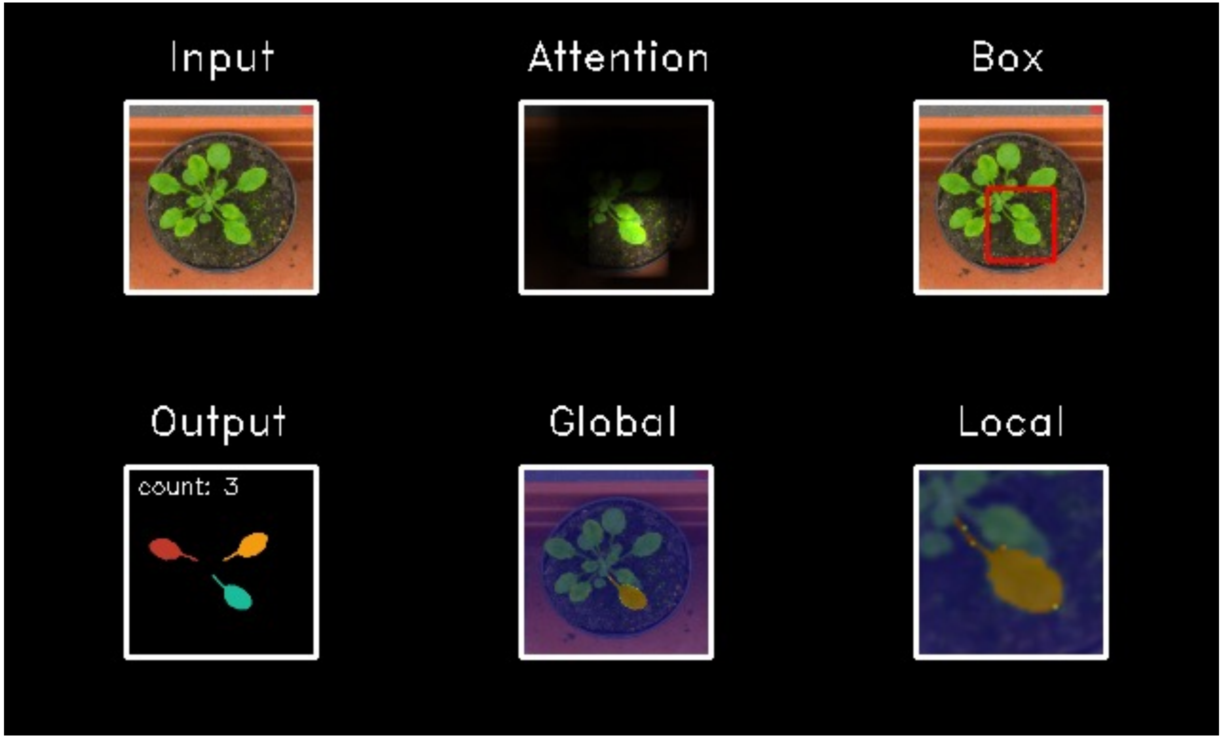
Demo

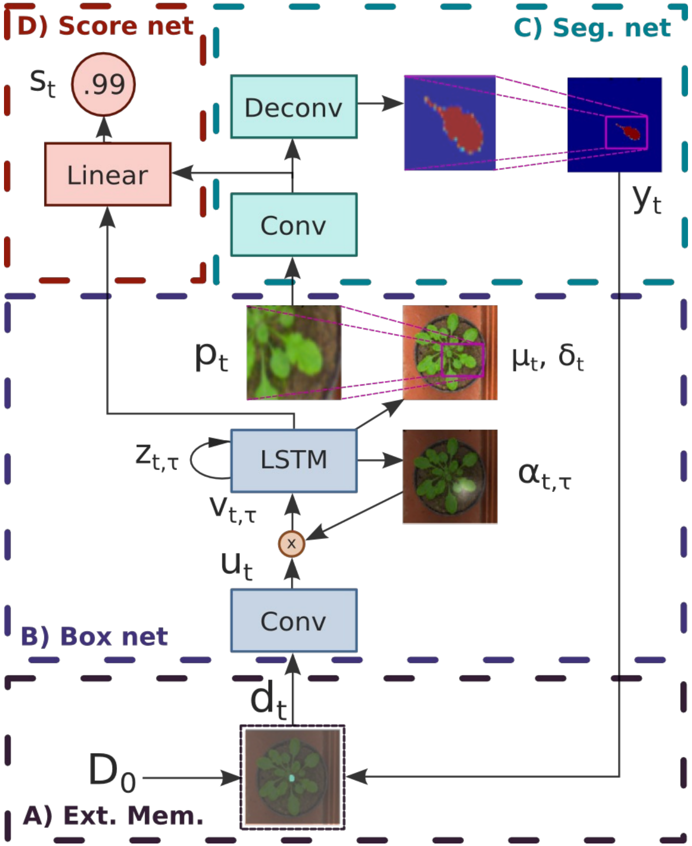




[Ren & Zemel, 2017]

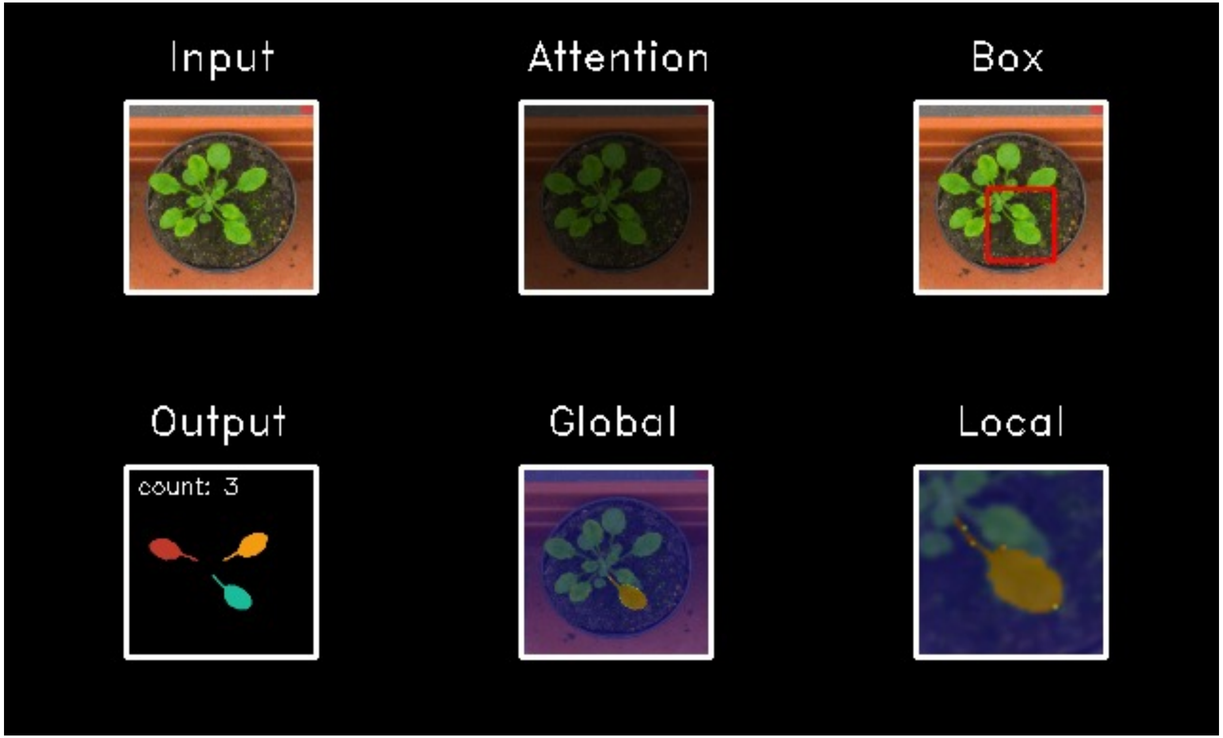
Demo

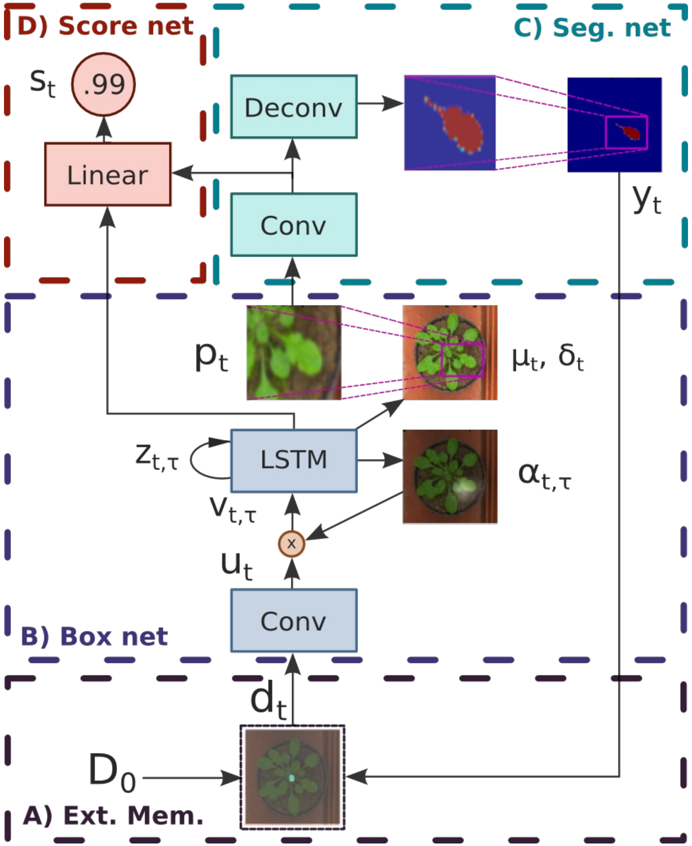




[Ren & Zemel, 2017]

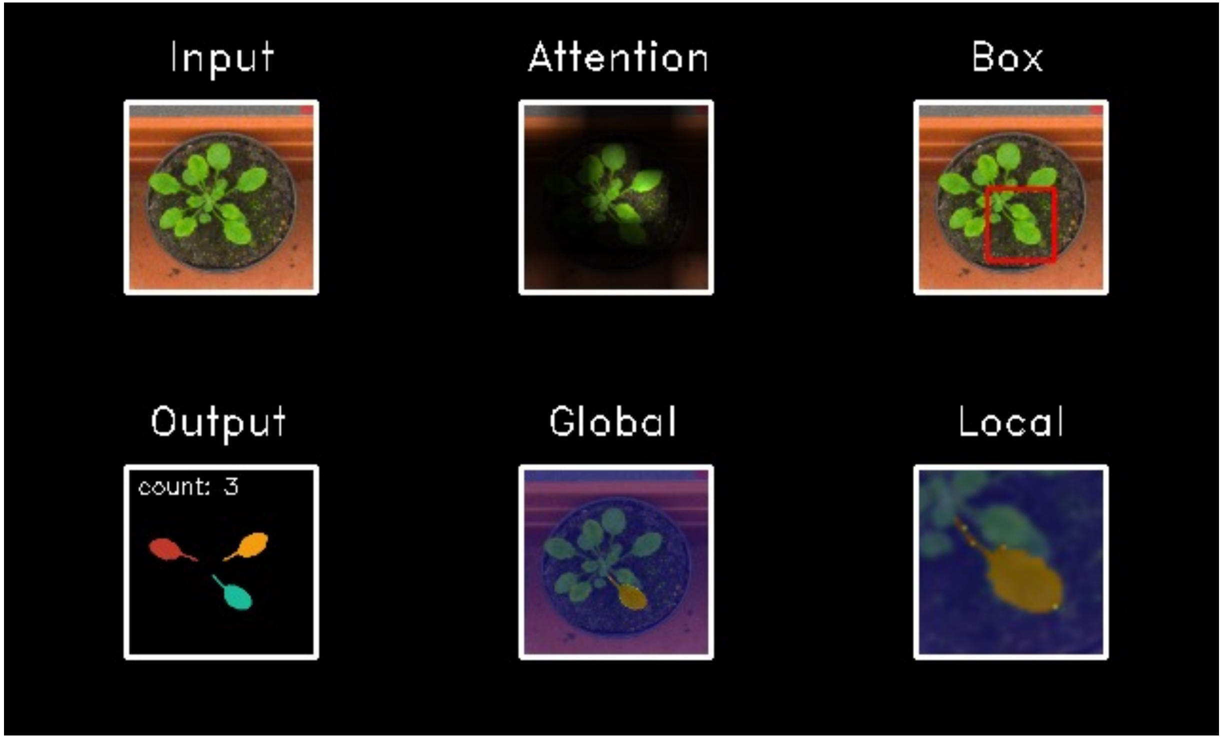
Demo

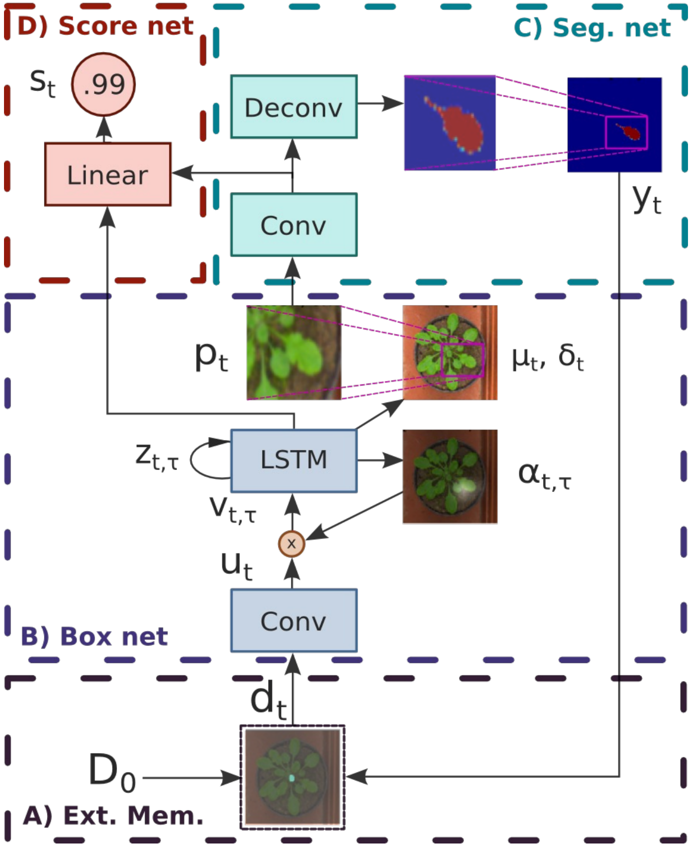




[Ren & Zemel, 2017]

Demo

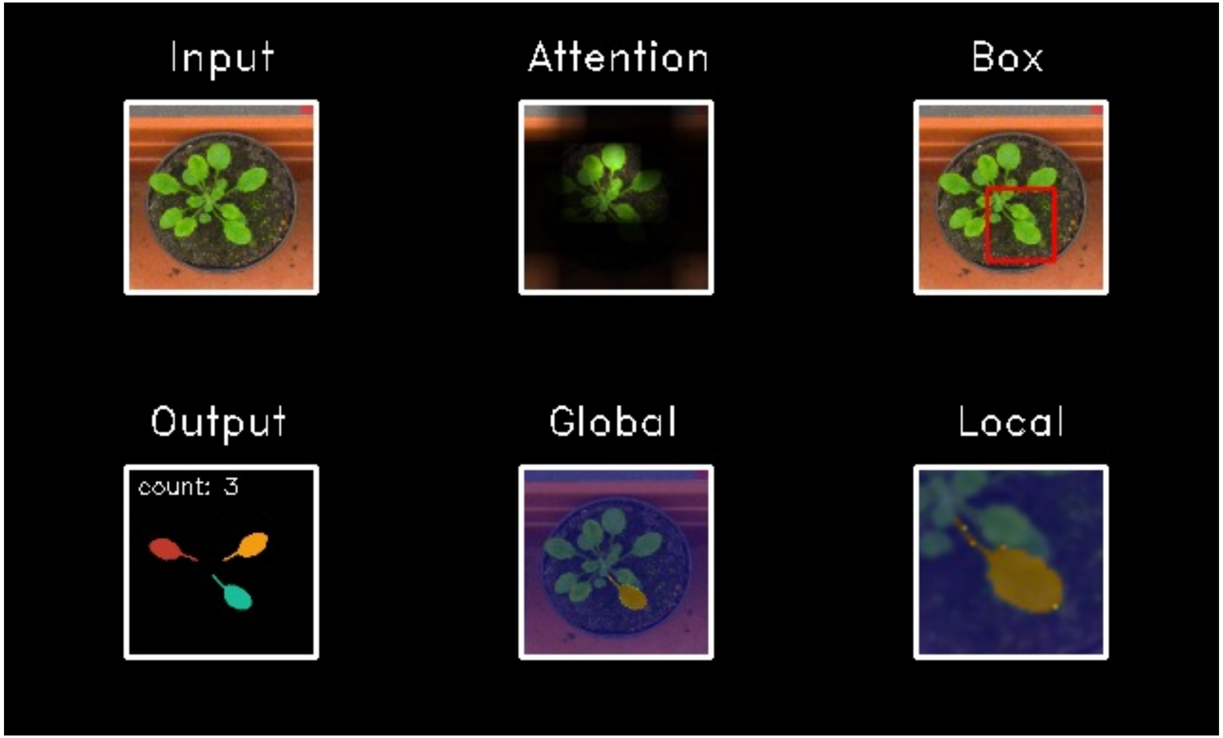


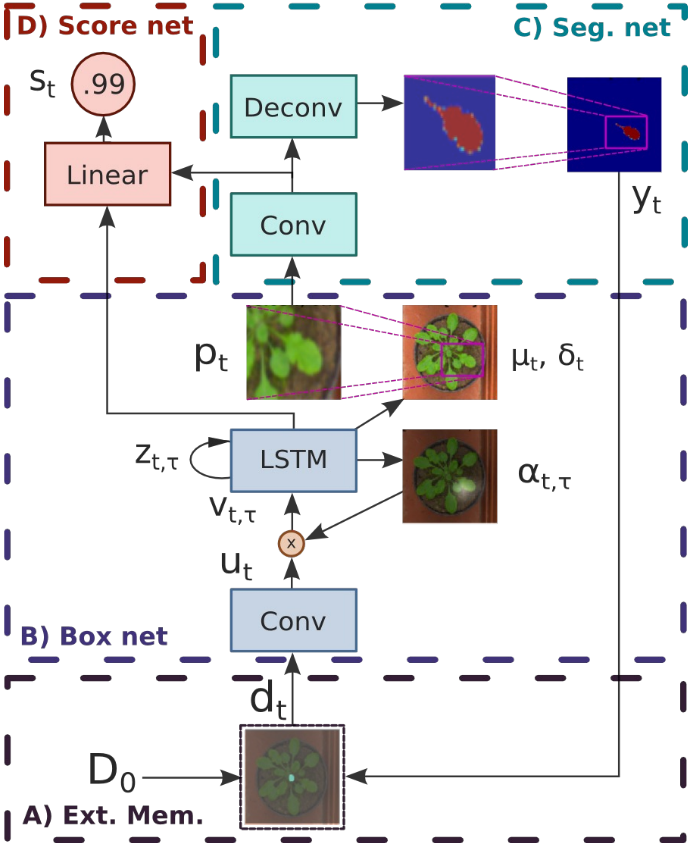


[Ren & Zemel, 2017]

NYU

Demo

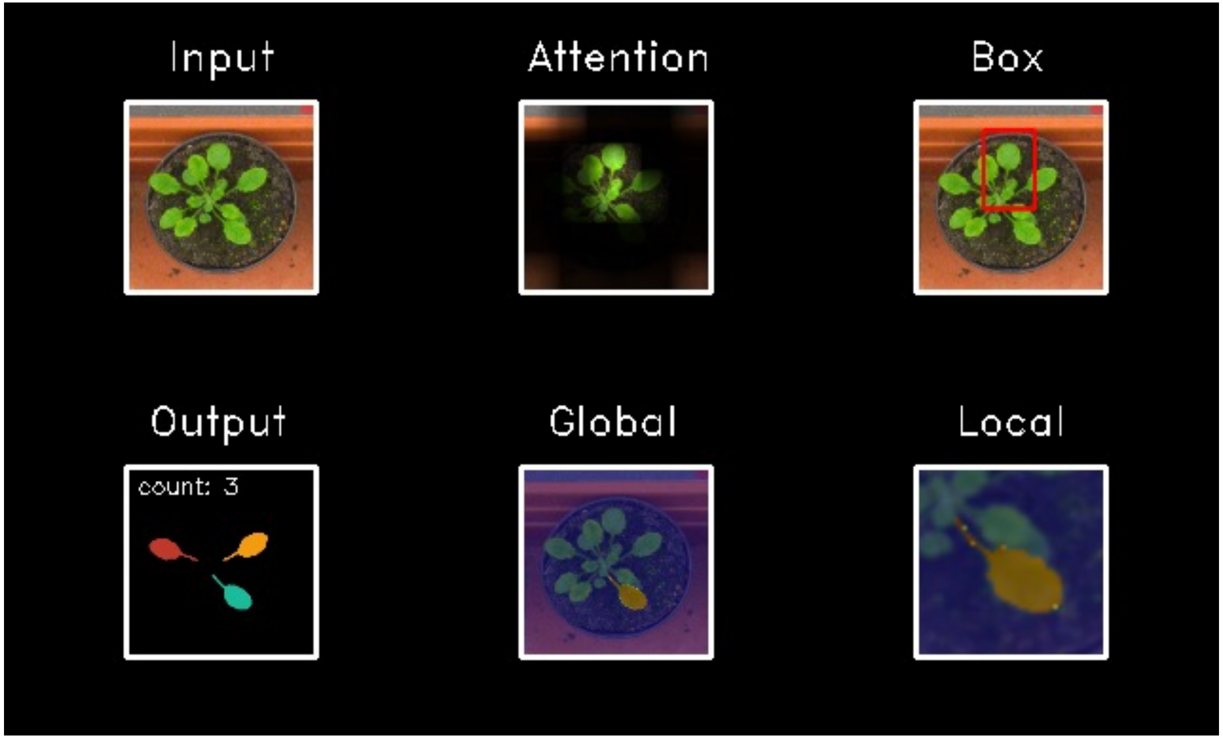


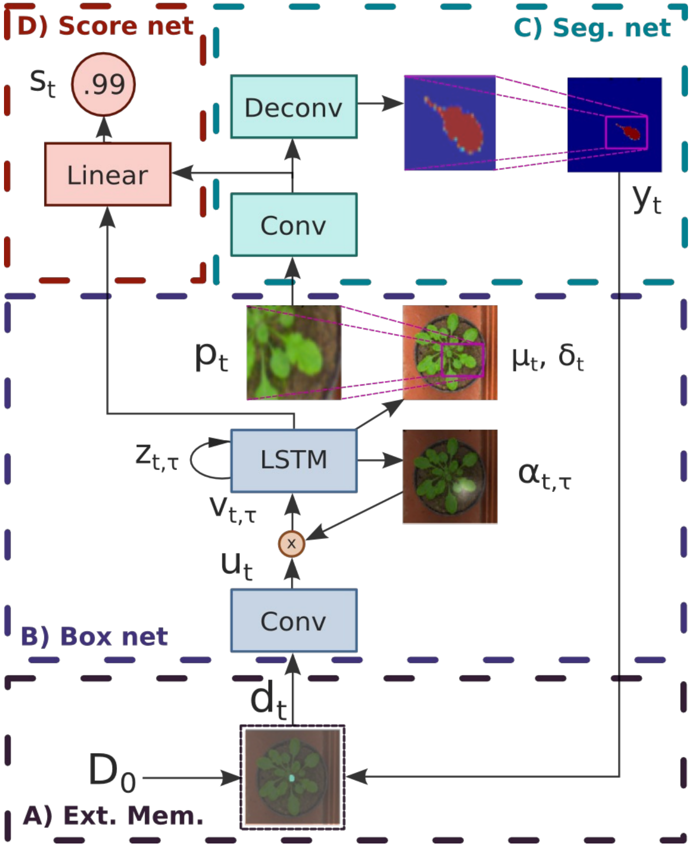


[Ren & Zemel, 2017]

NYU

Demo

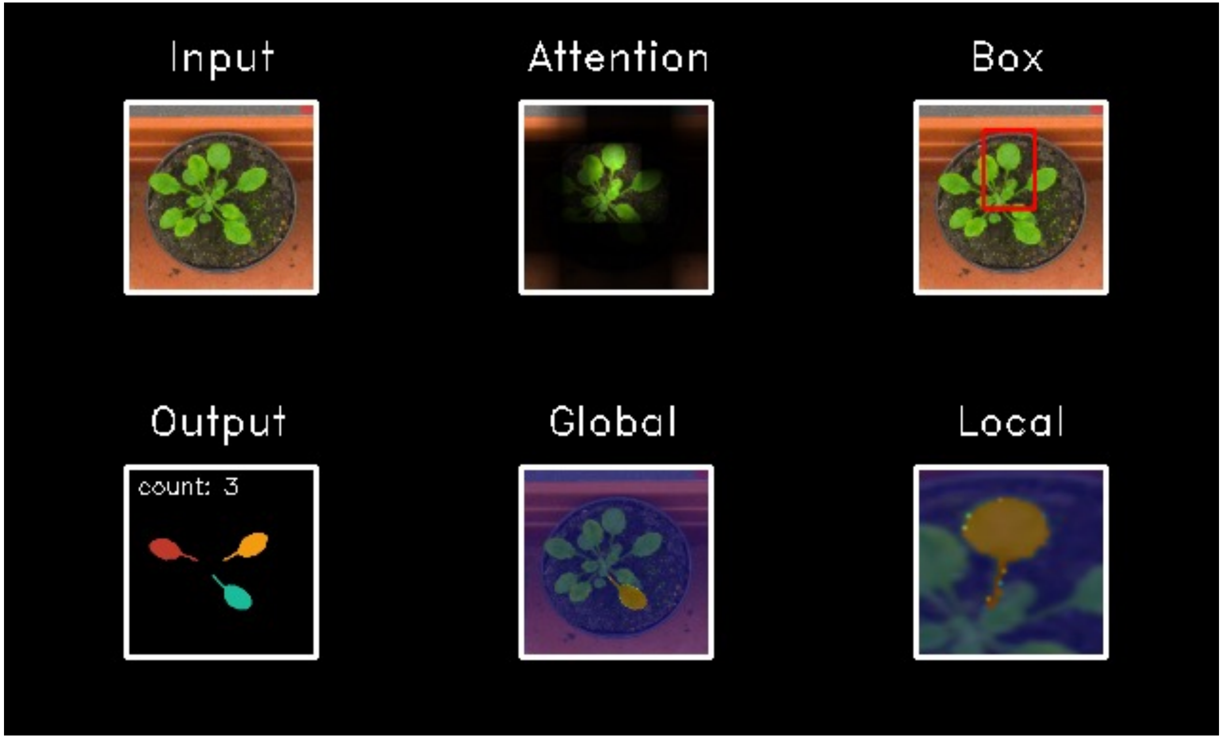


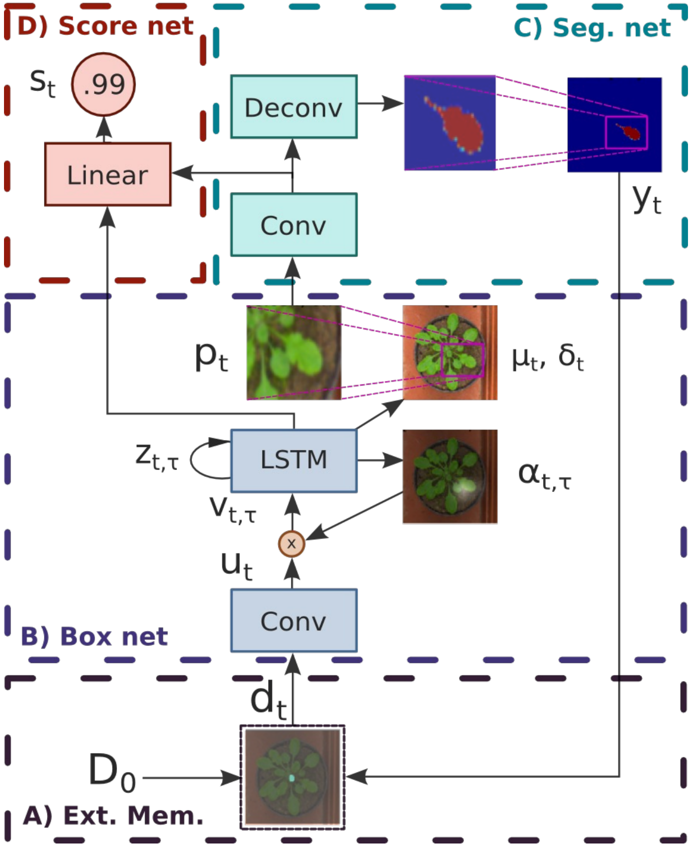


[Ren & Zemel, 2017]

NYU

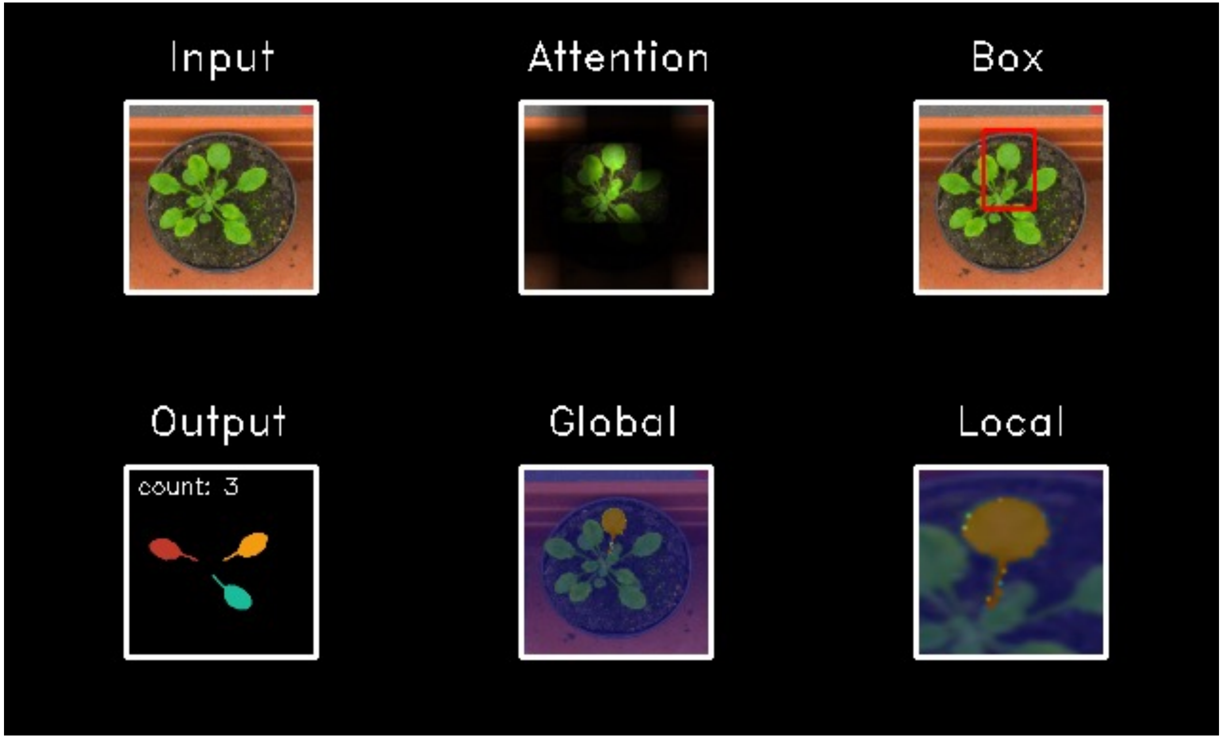
Demo

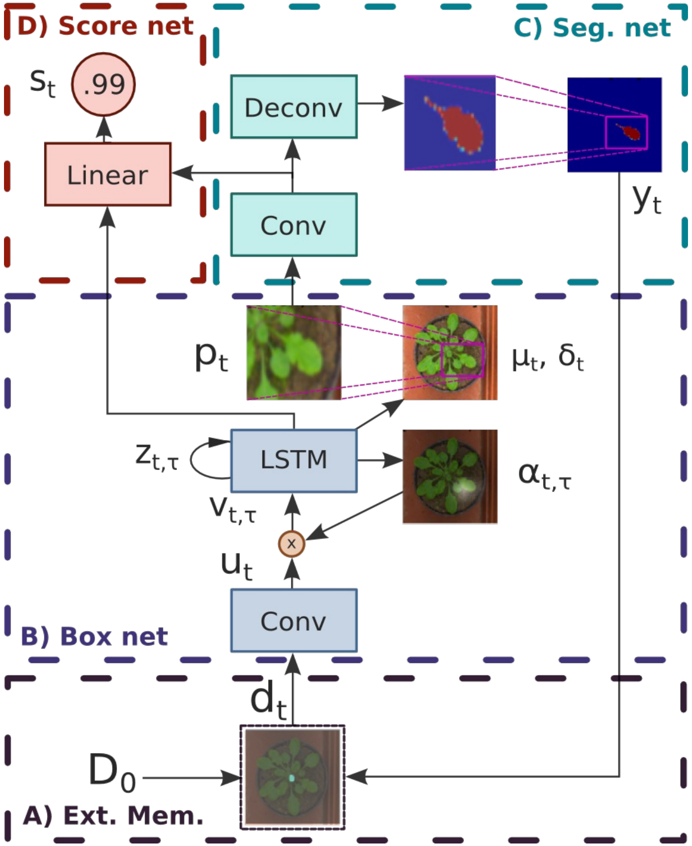




[Ren & Zemel, 2017]

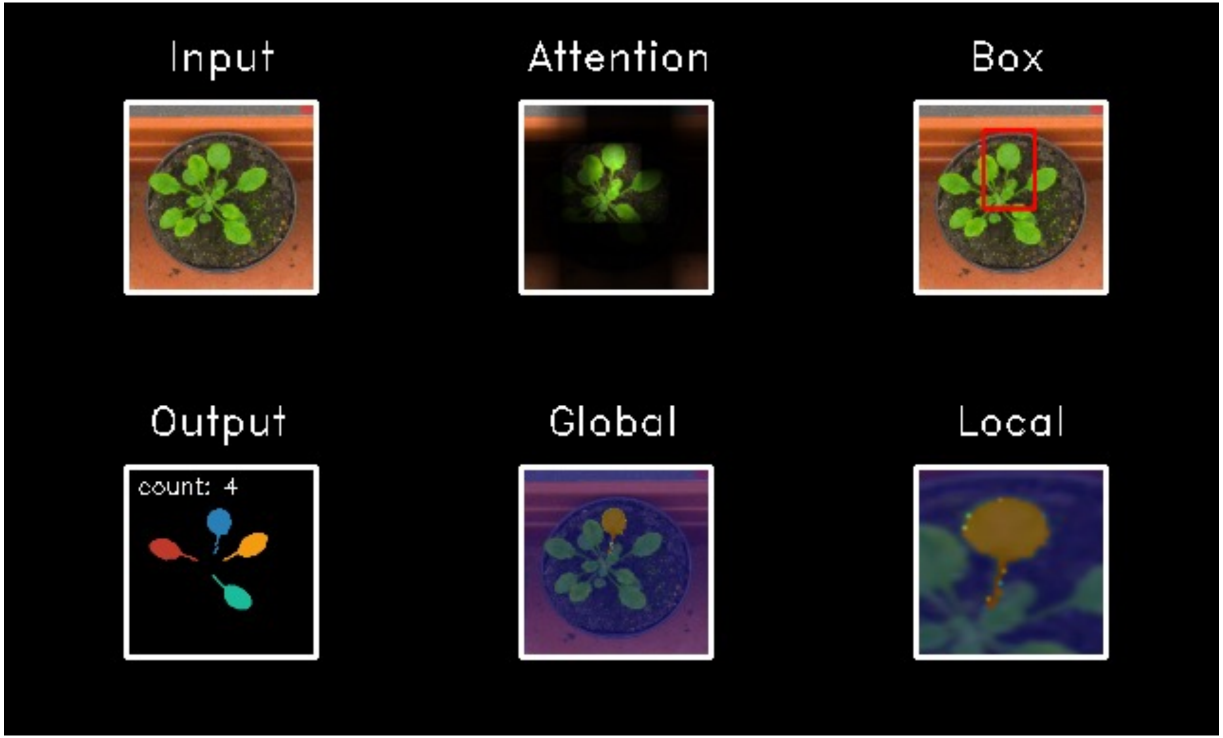
Demo

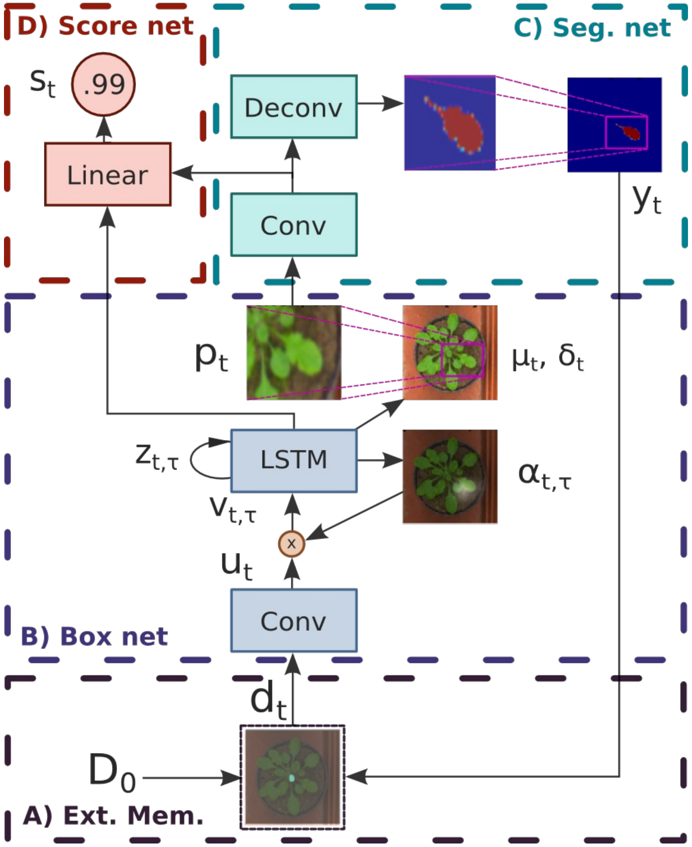




[Ren & Zemel, 2017]

Demo

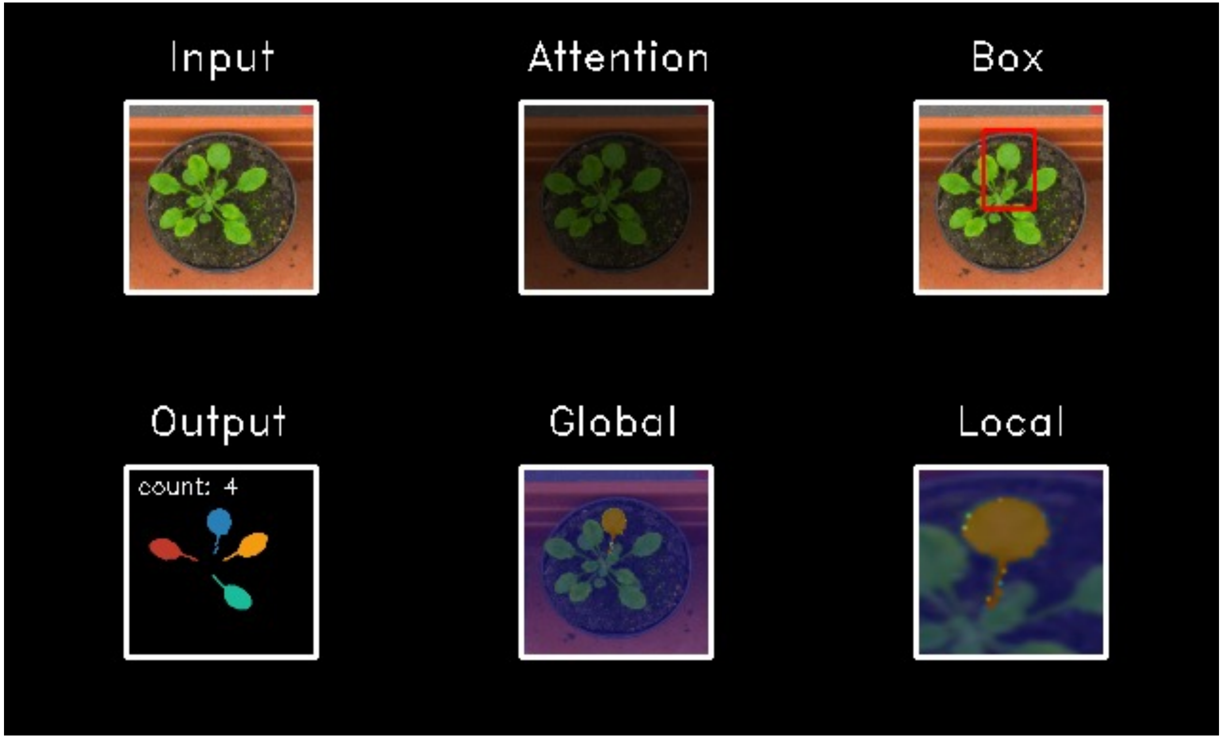


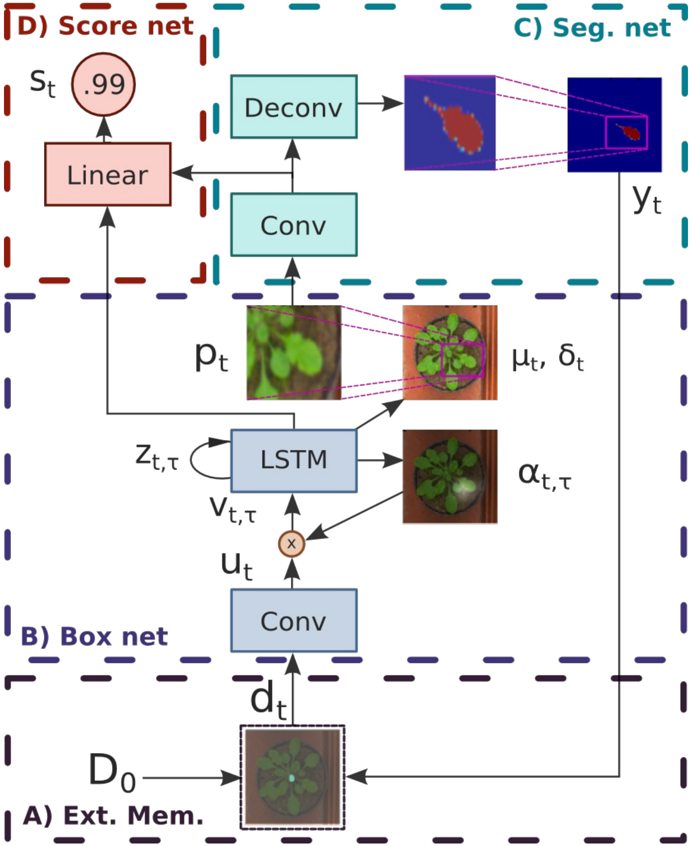


[Ren & Zemel, 2017]

NYU

Demo

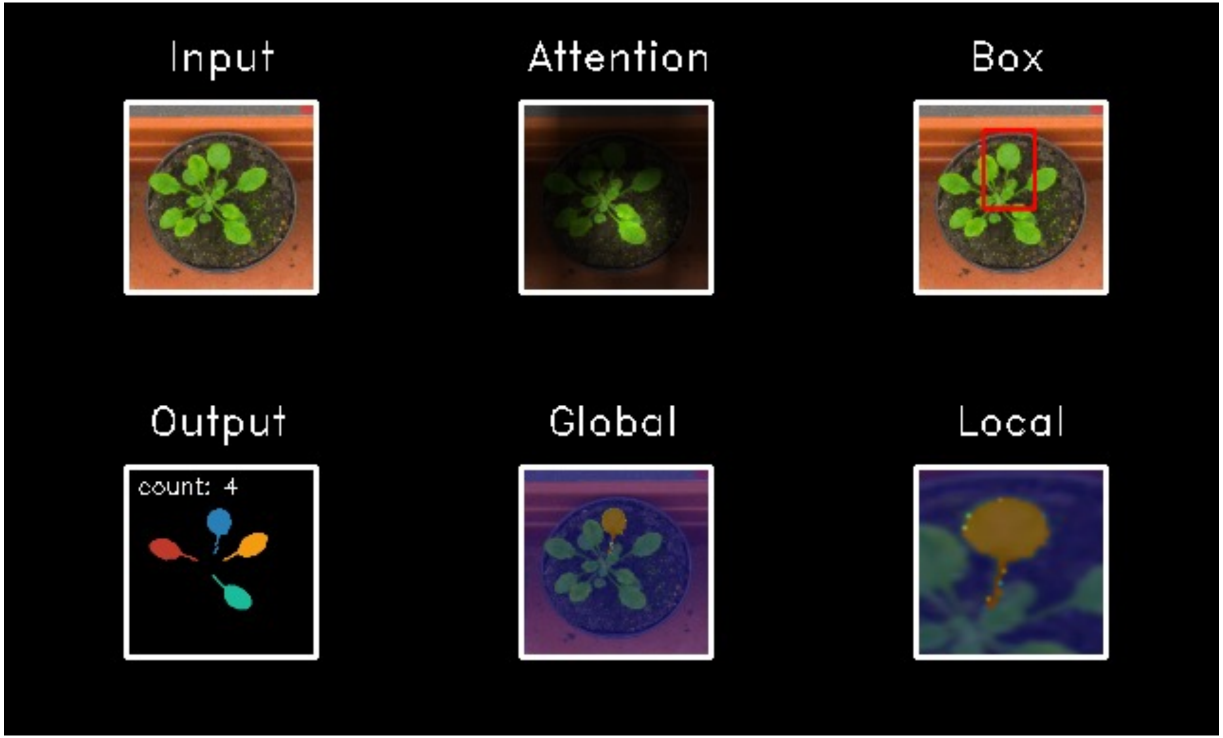


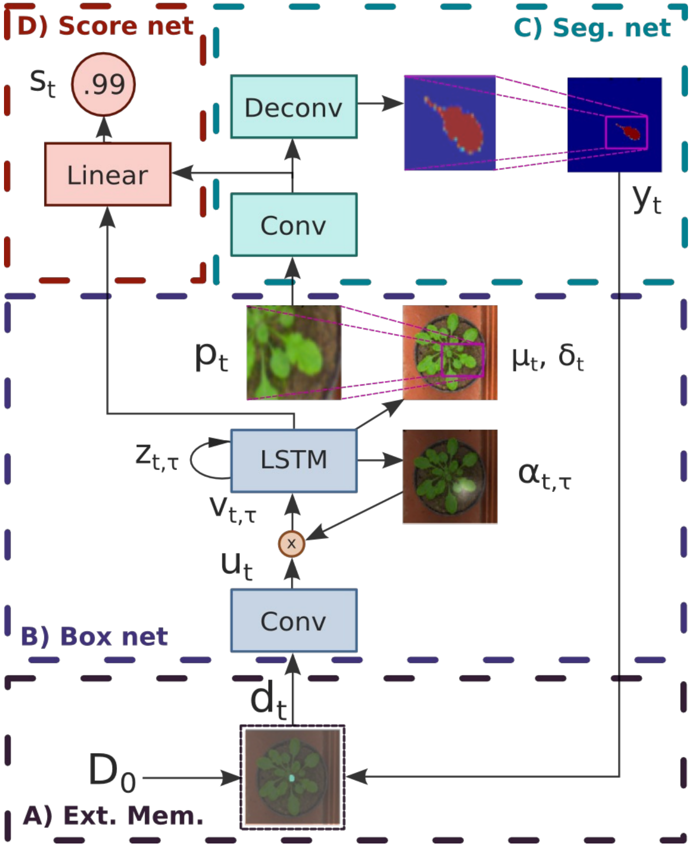


[Ren & Zemel, 2017]

NYU

Demo

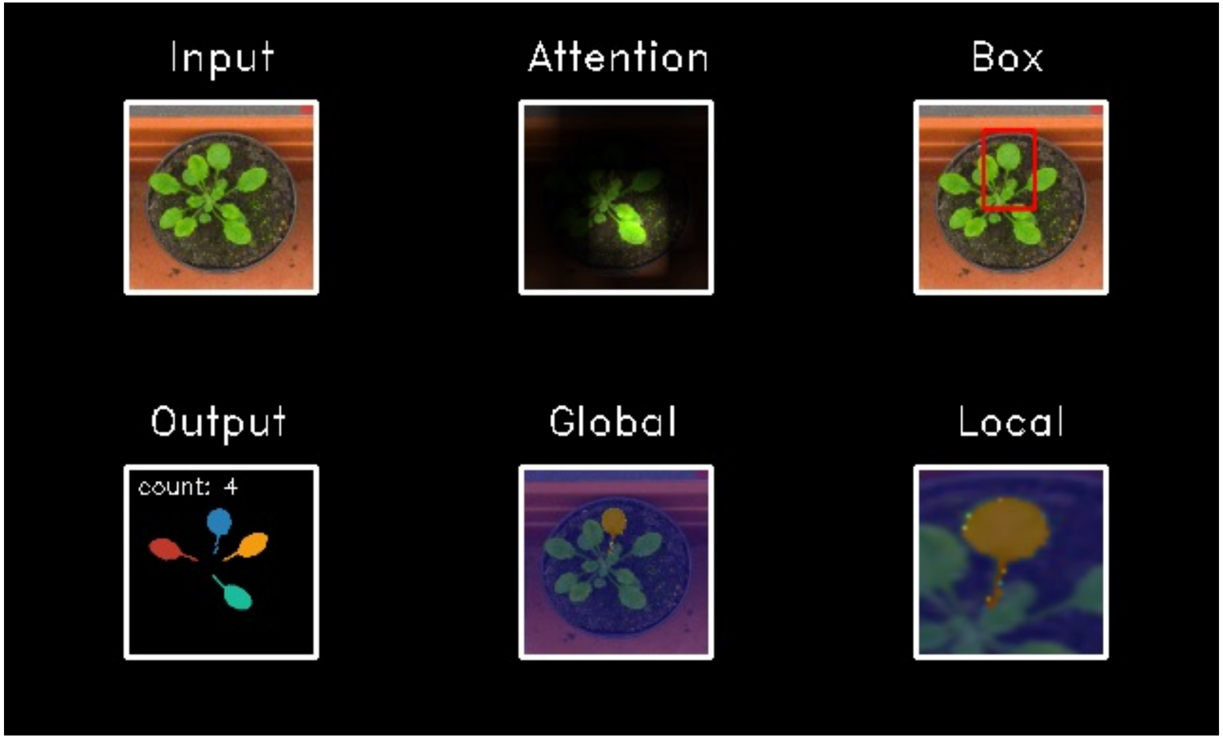


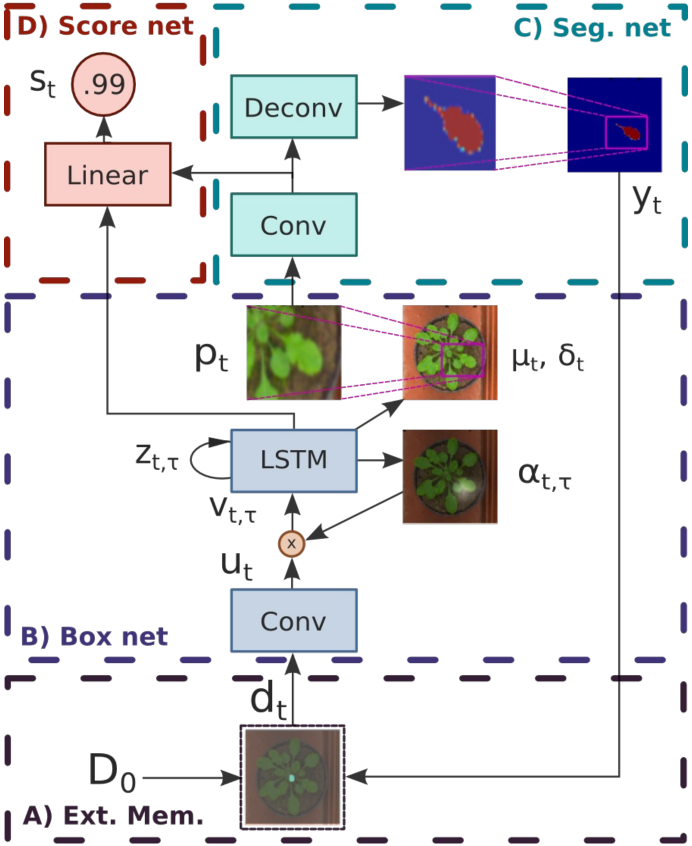


[Ren & Zemel, 2017]

NYU

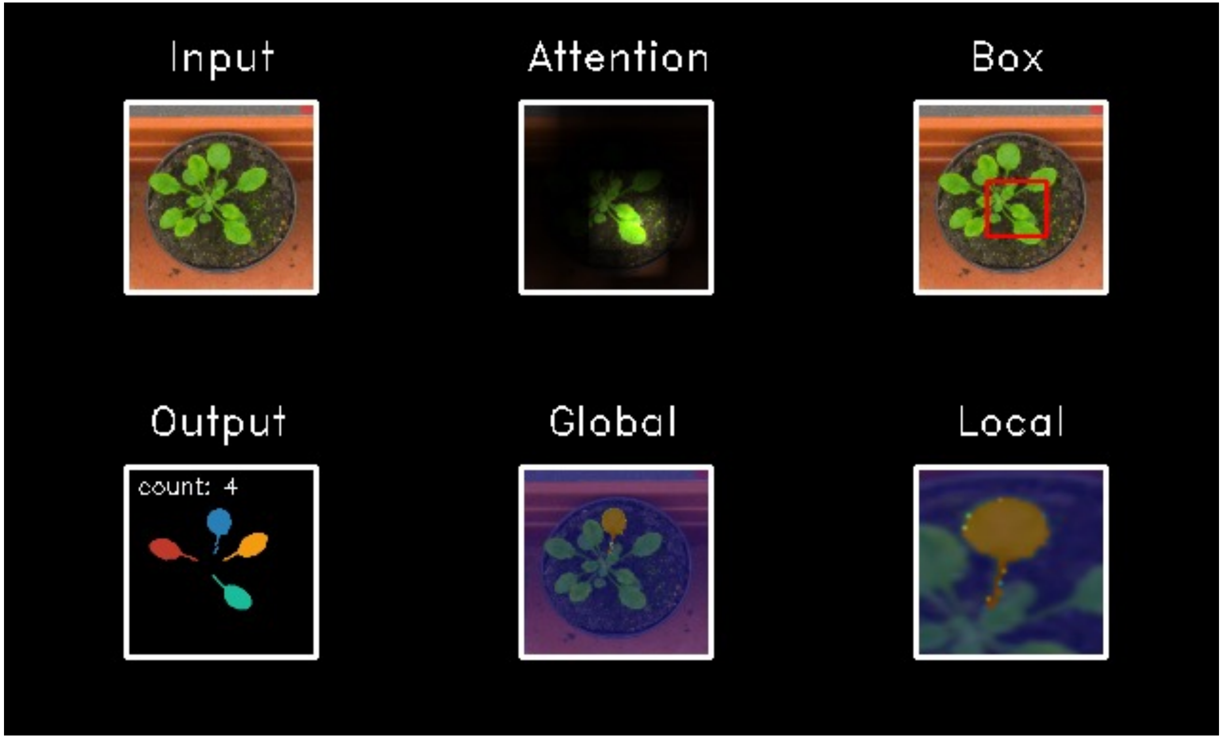
Demo

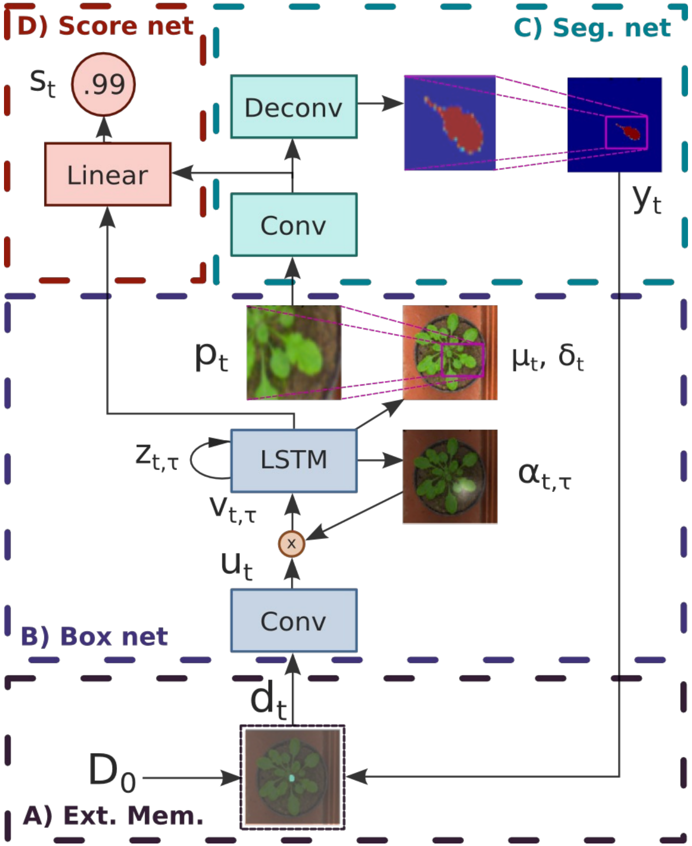




[Ren & Zemel, 2017]

Demo

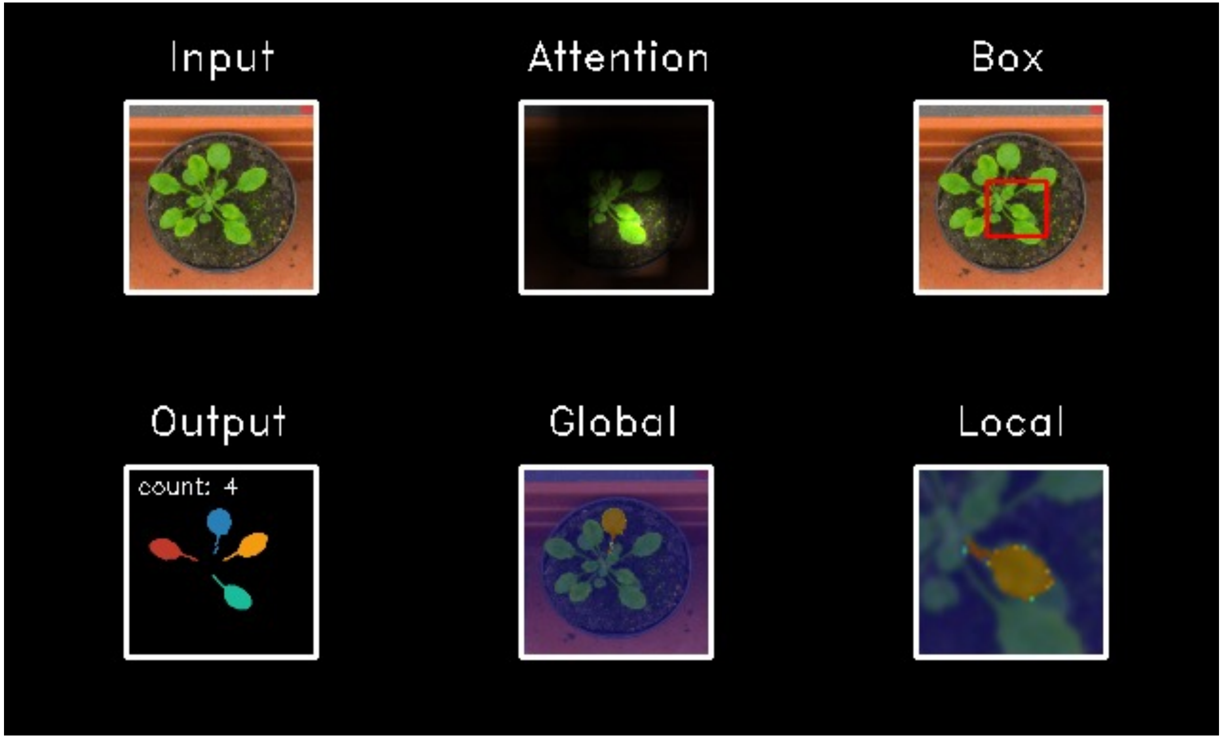


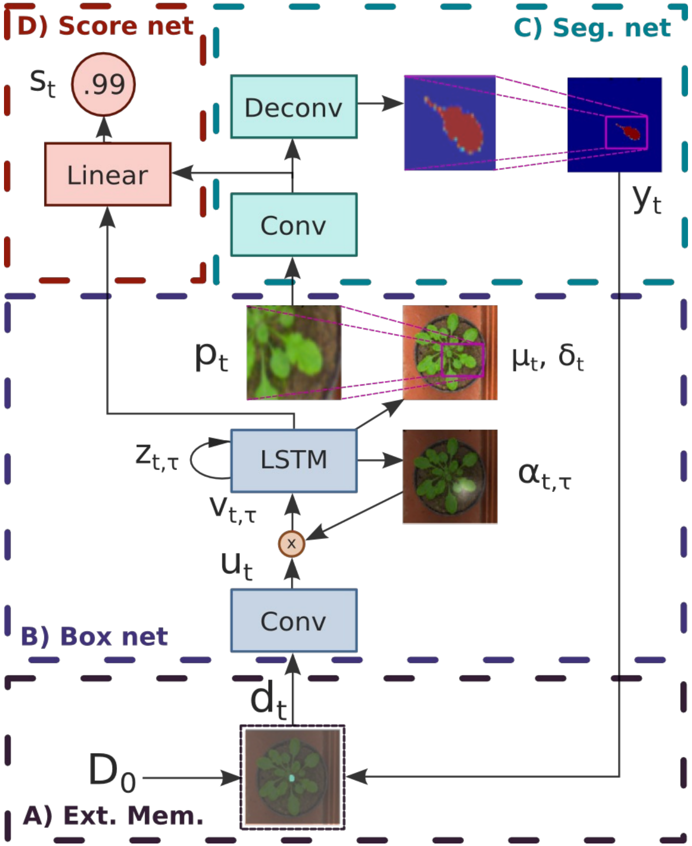


[Ren & Zemel, 2017]

NYU

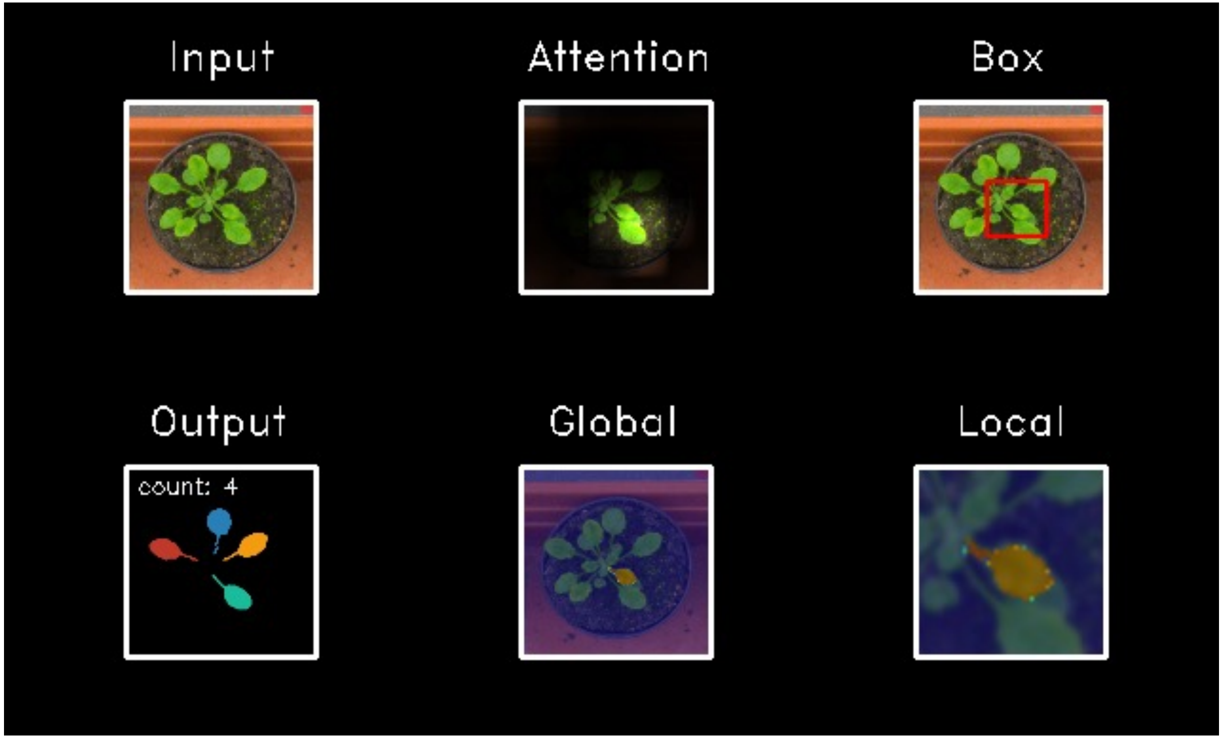
Demo

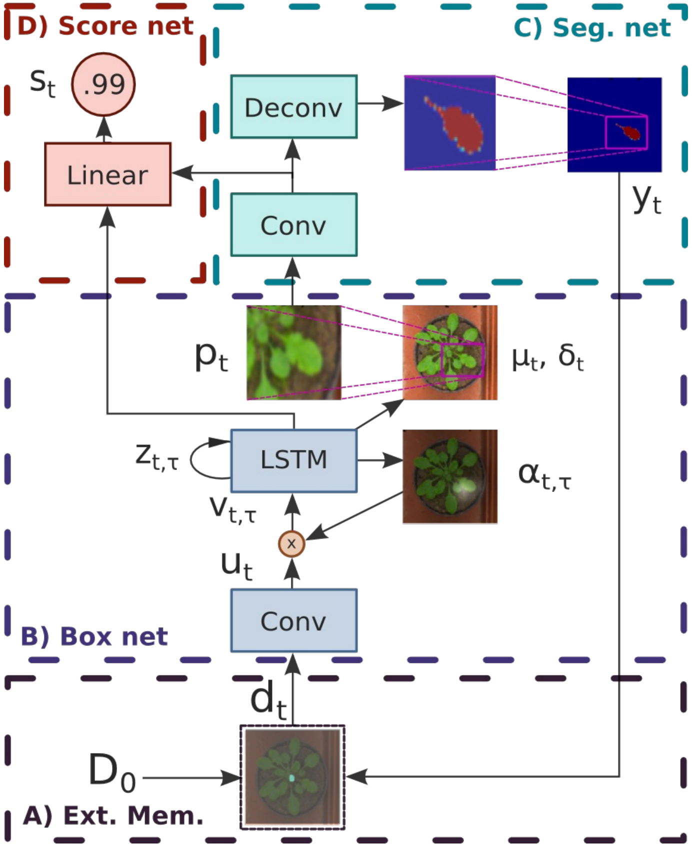




[Ren & Zemel, 2017]

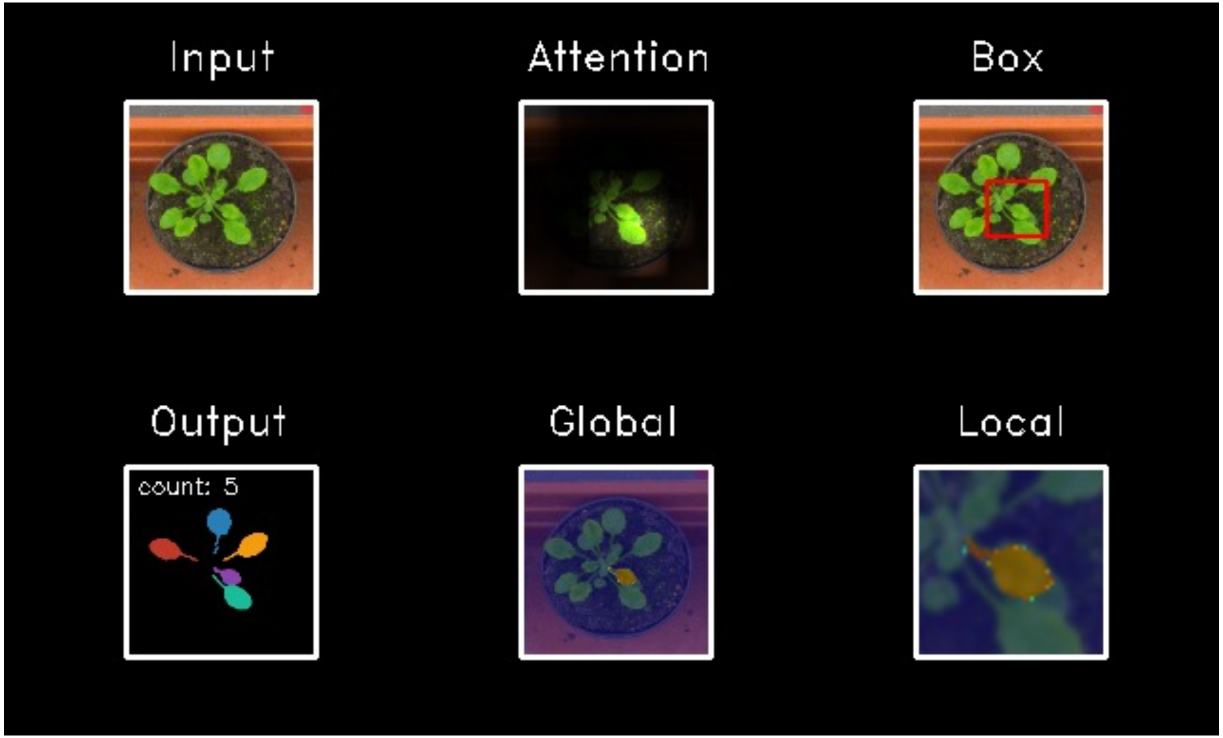
Demo

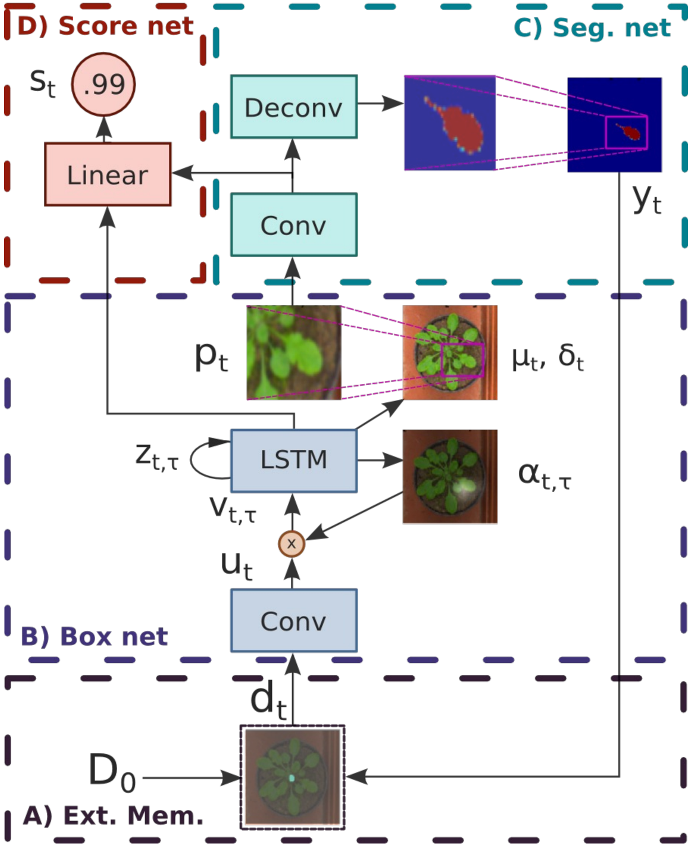




[Ren & Zemel, 2017]

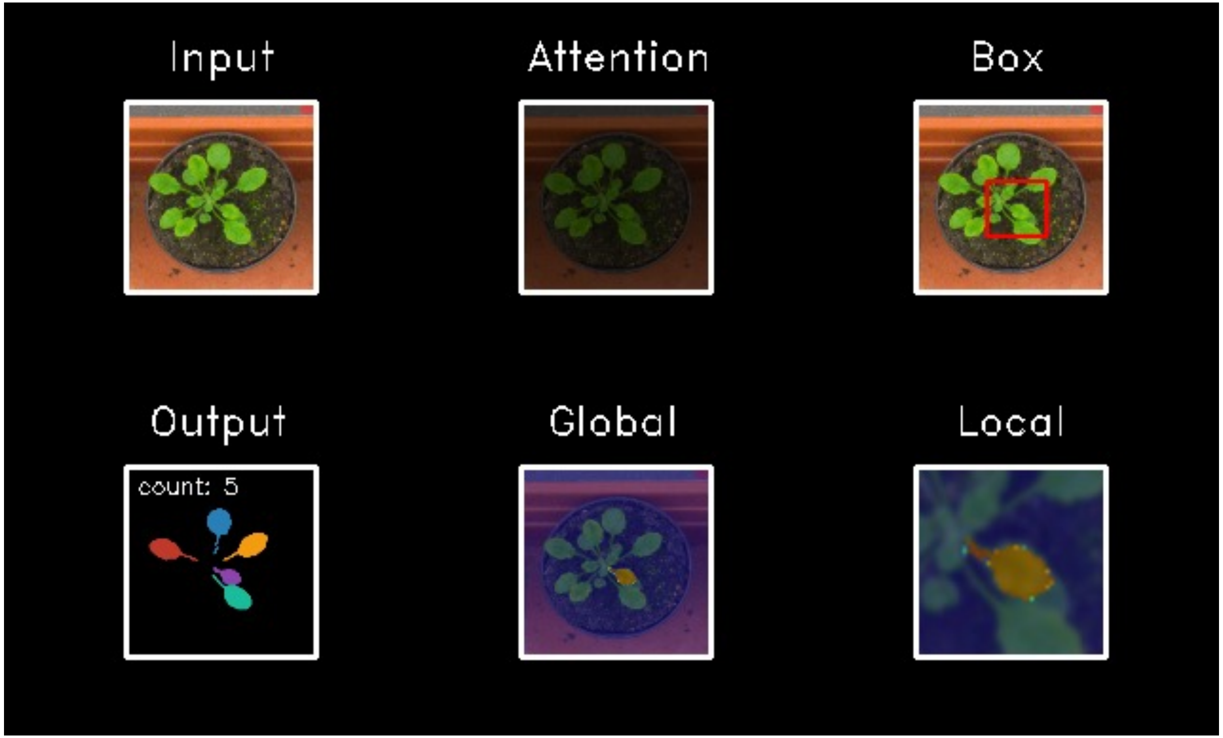
Demo

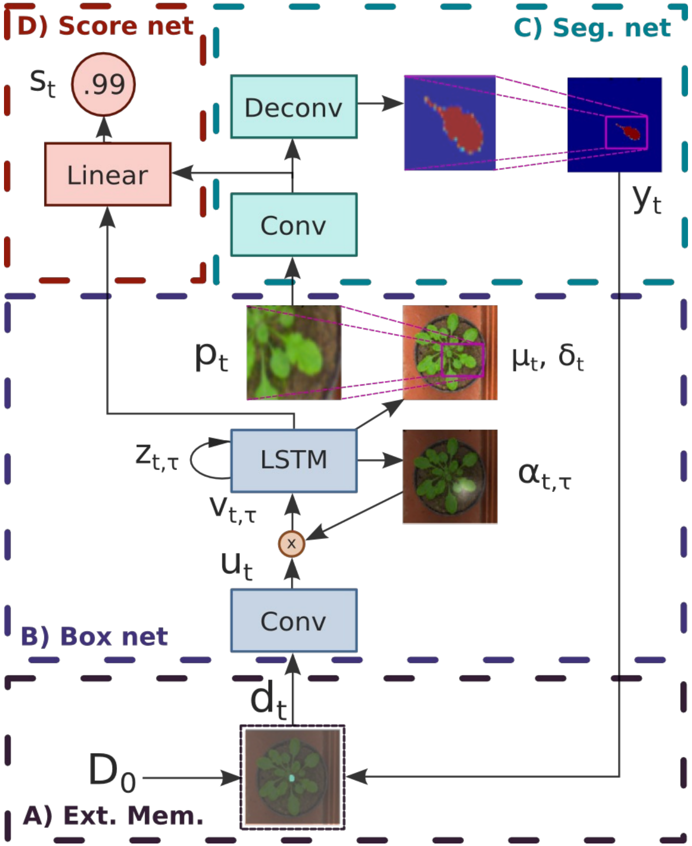




[Ren & Zemel, 2017]

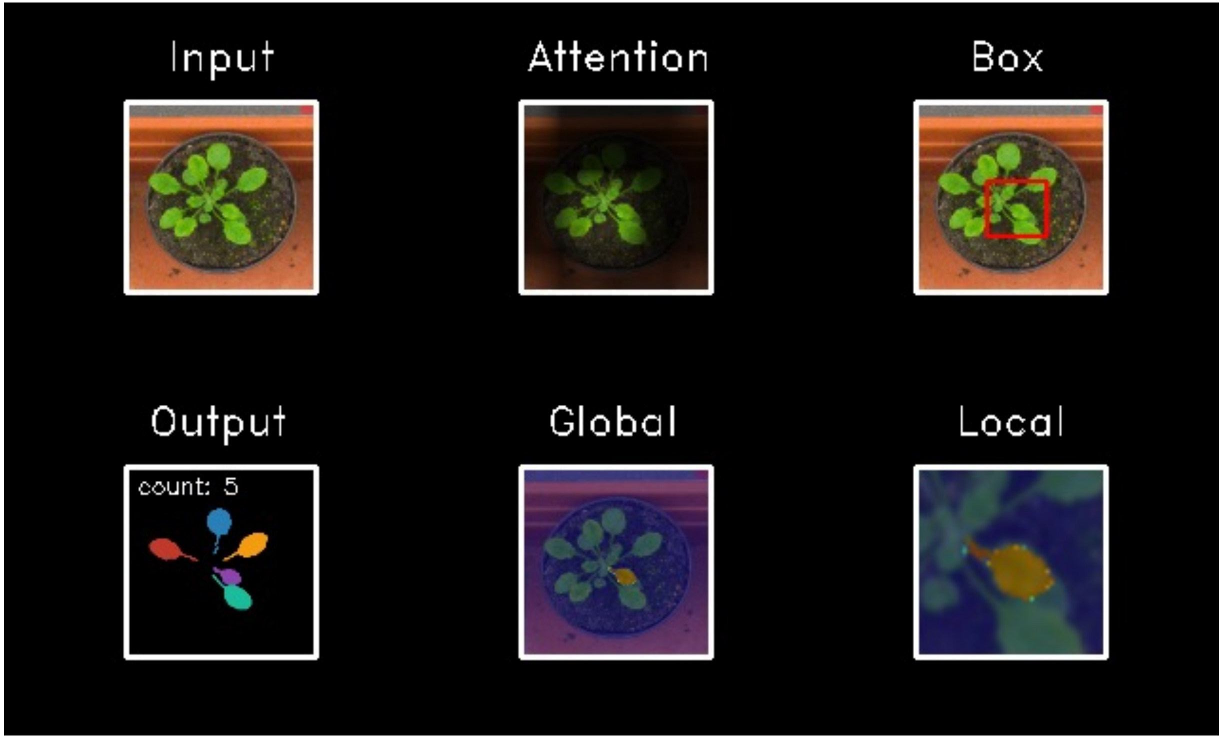
Demo

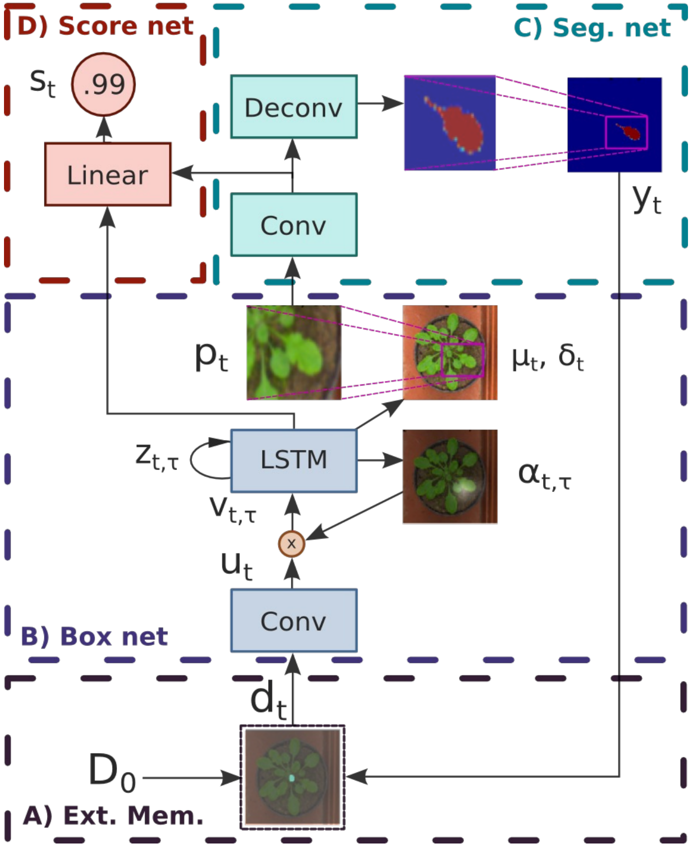




[Ren & Zemel, 2017]

Demo

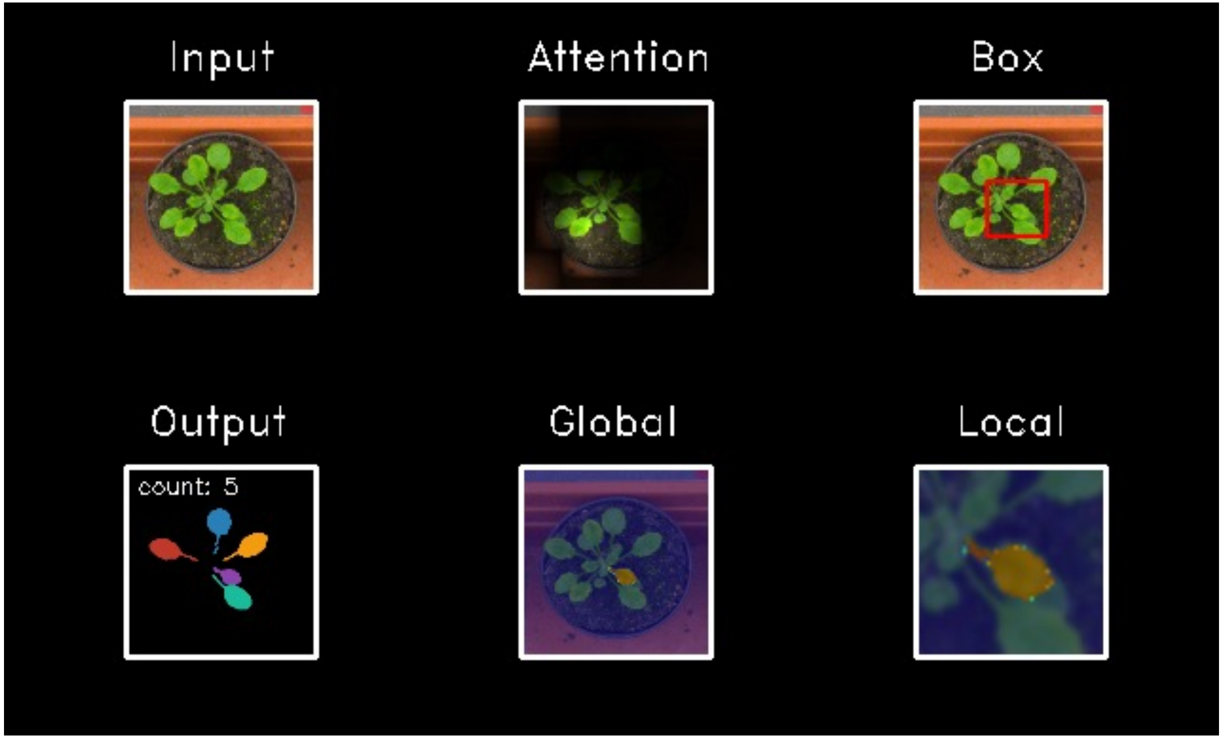


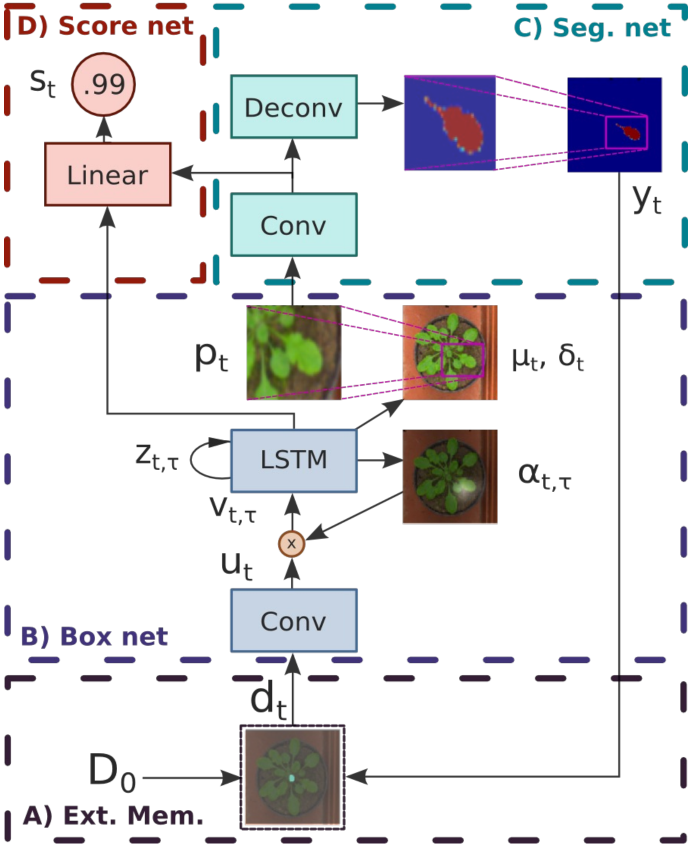


[Ren & Zemel, 2017]

NYU

Demo

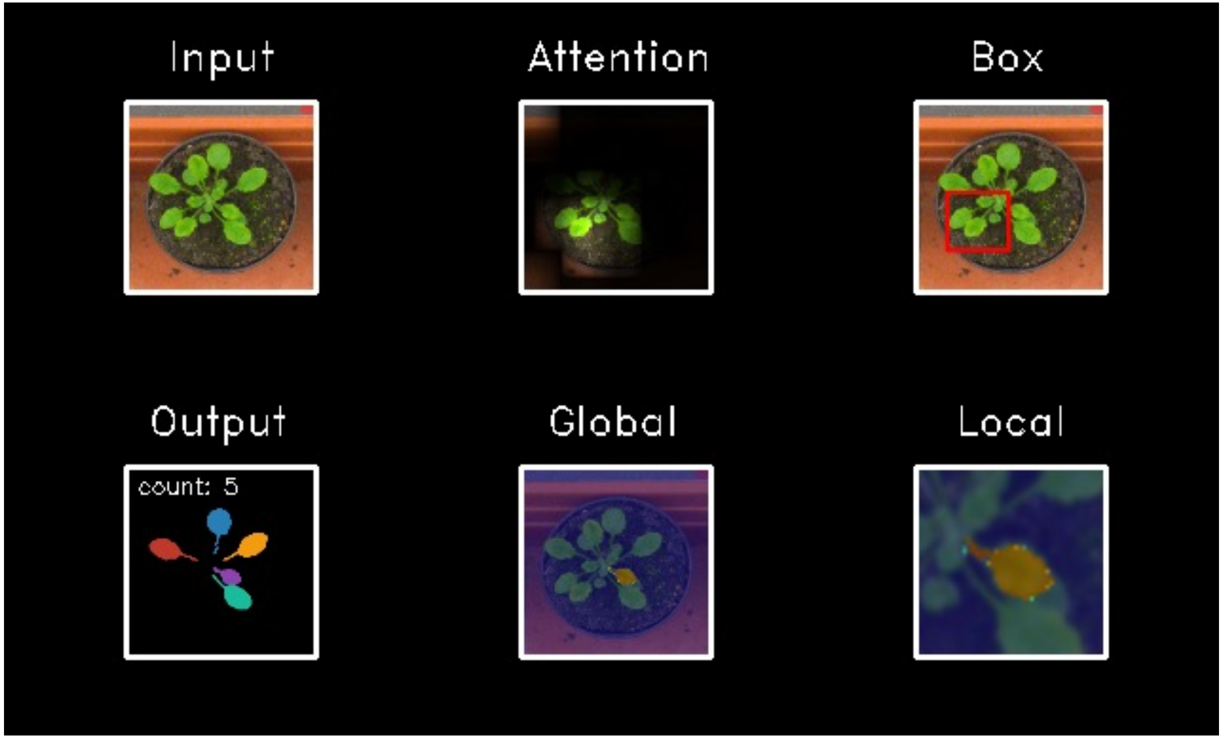


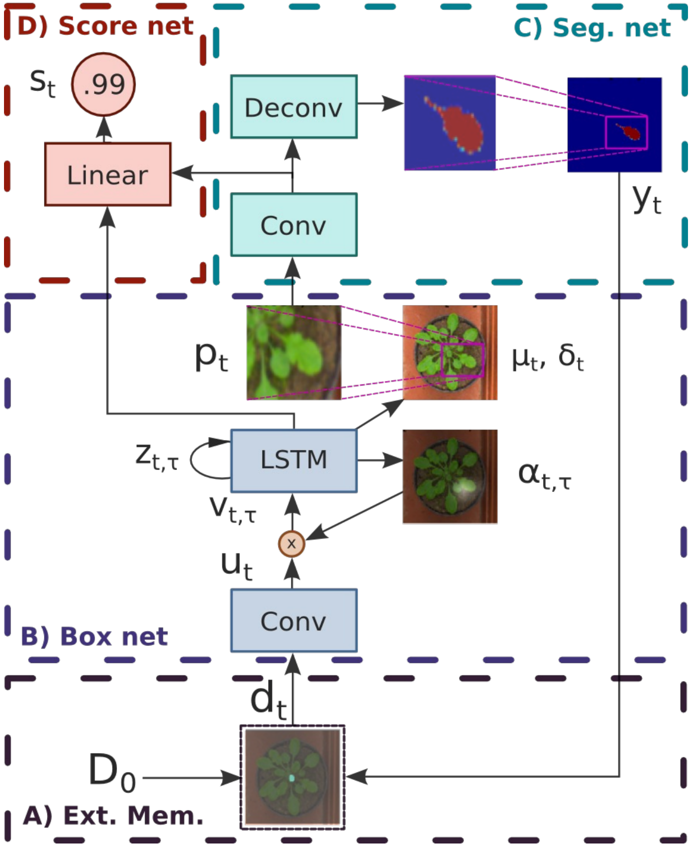


[Ren & Zemel, 2017]

NYU

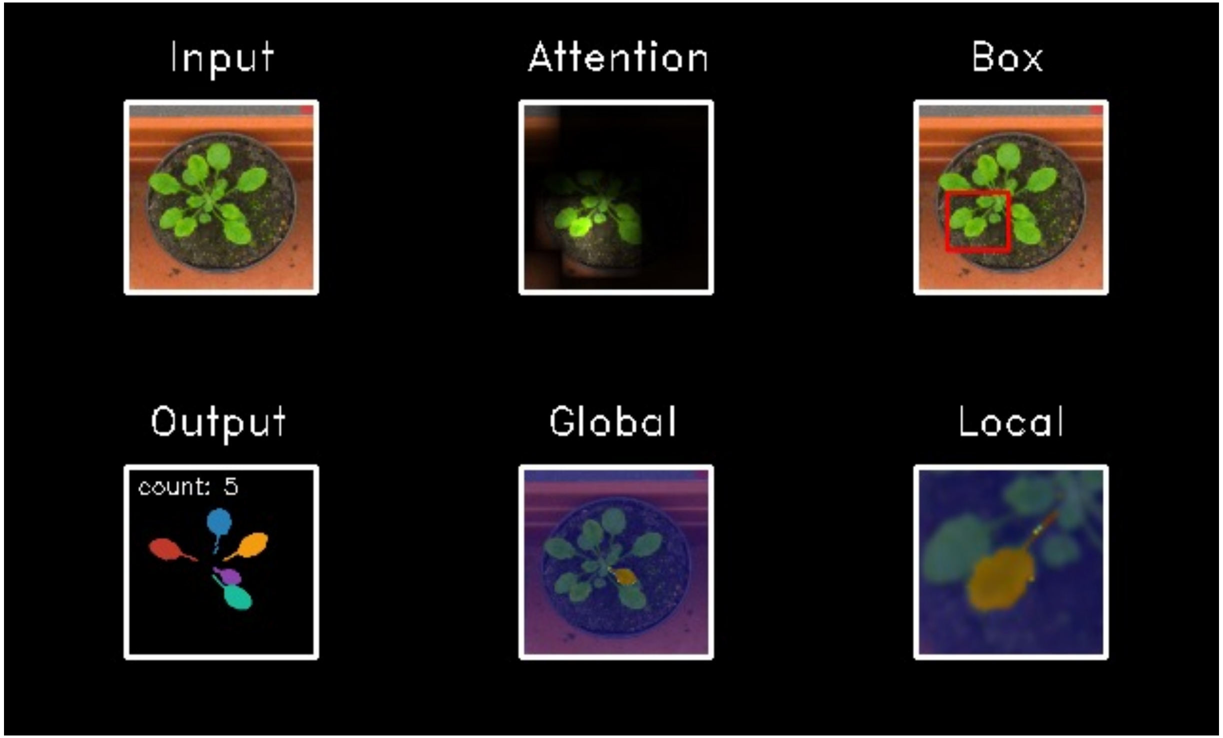
Demo

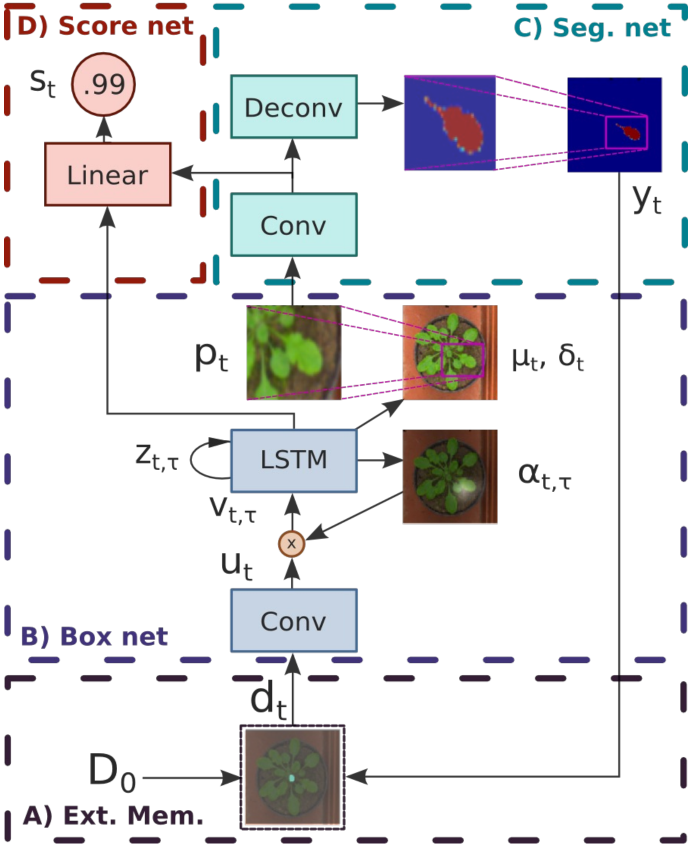




[Ren & Zemel, 2017]

Demo

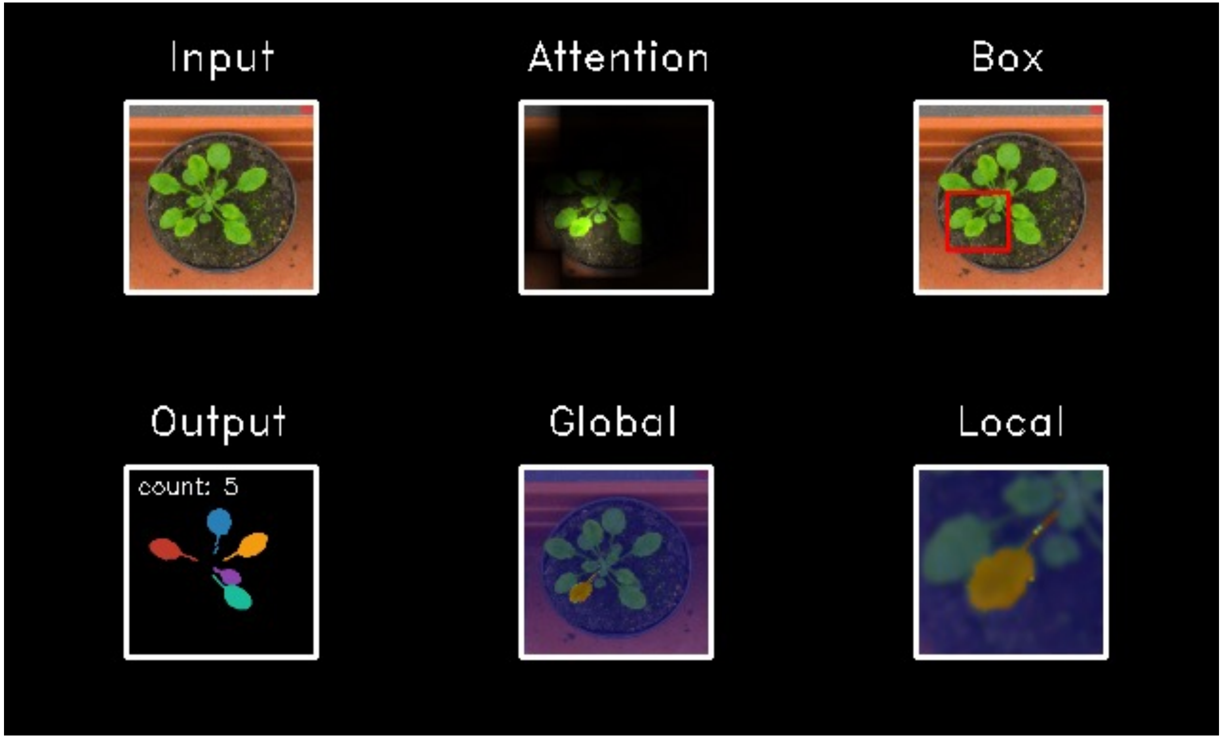


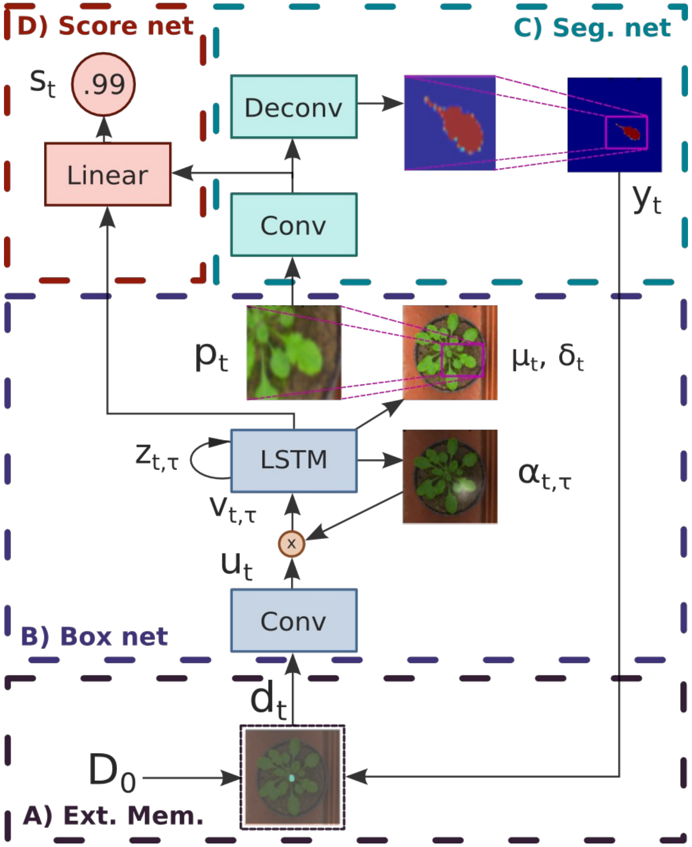


[Ren & Zemel, 2017]

NYU

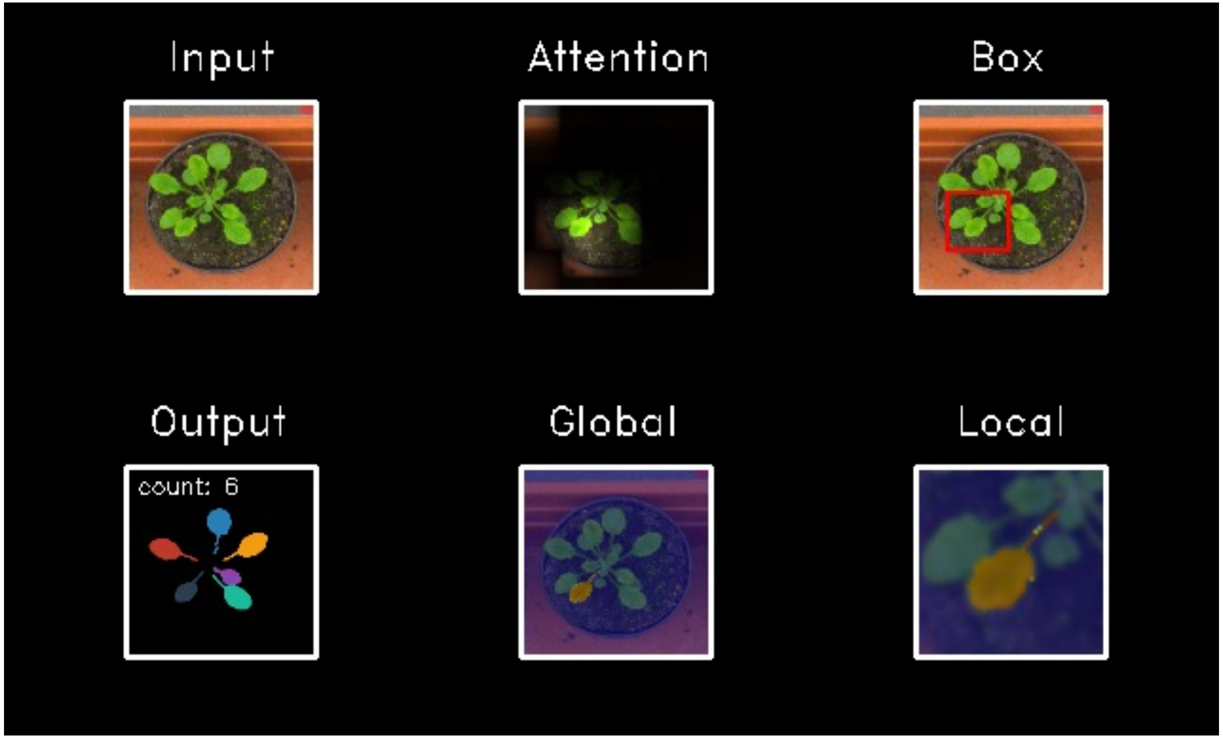
Demo

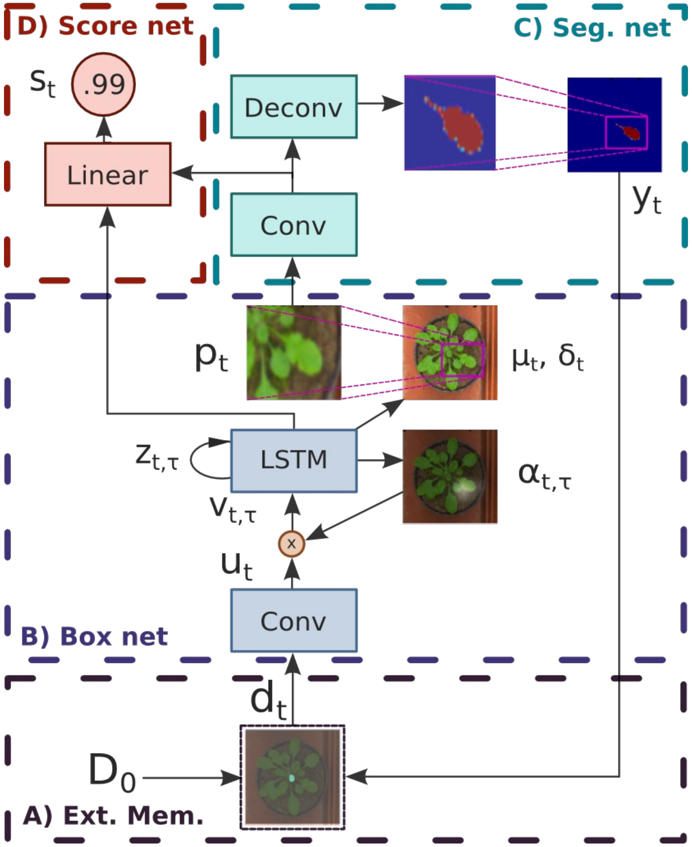




[Ren & Zemel, 2017]

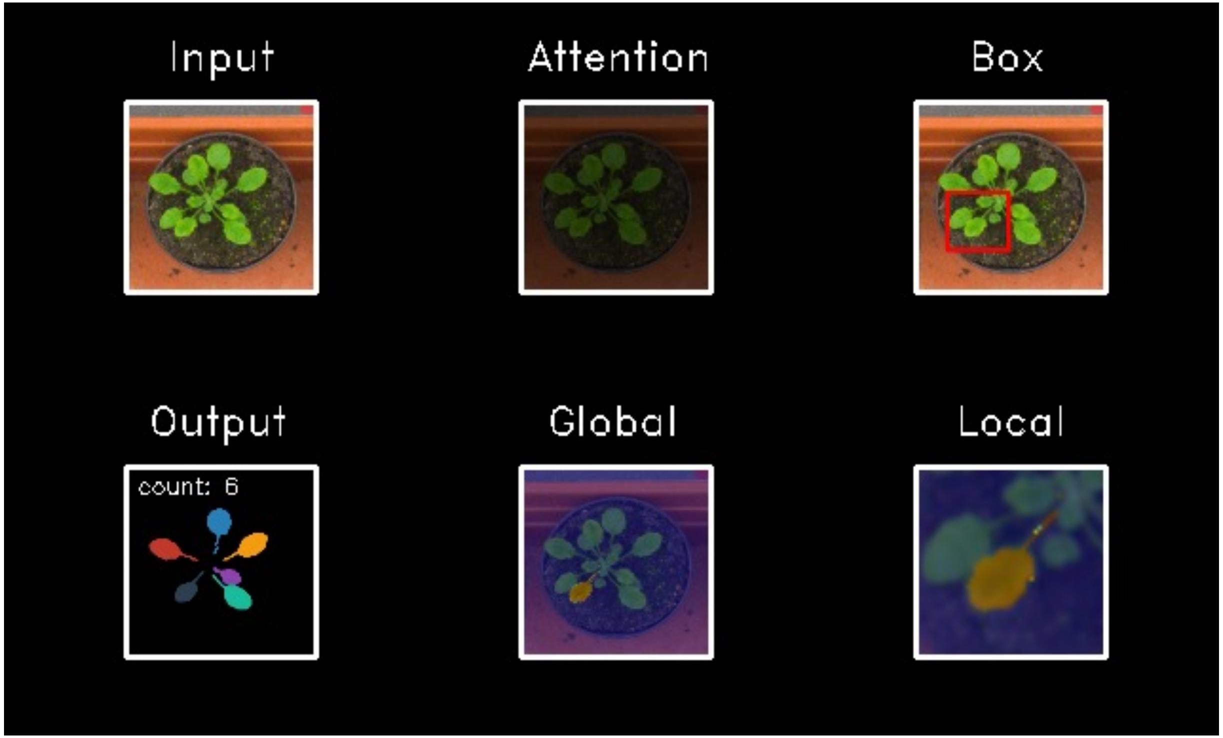
Demo

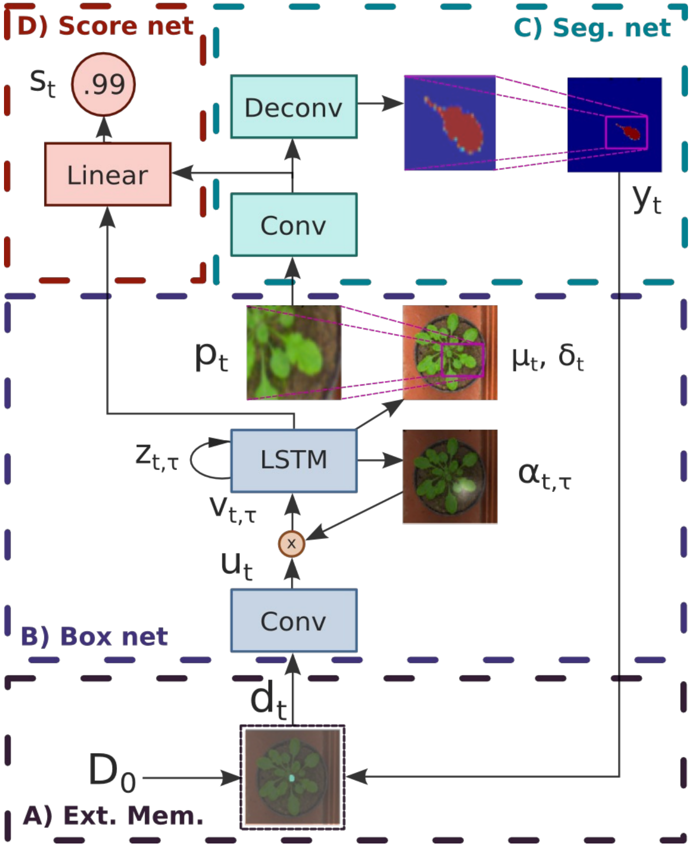




[Ren & Zemel, 2017]

Demo

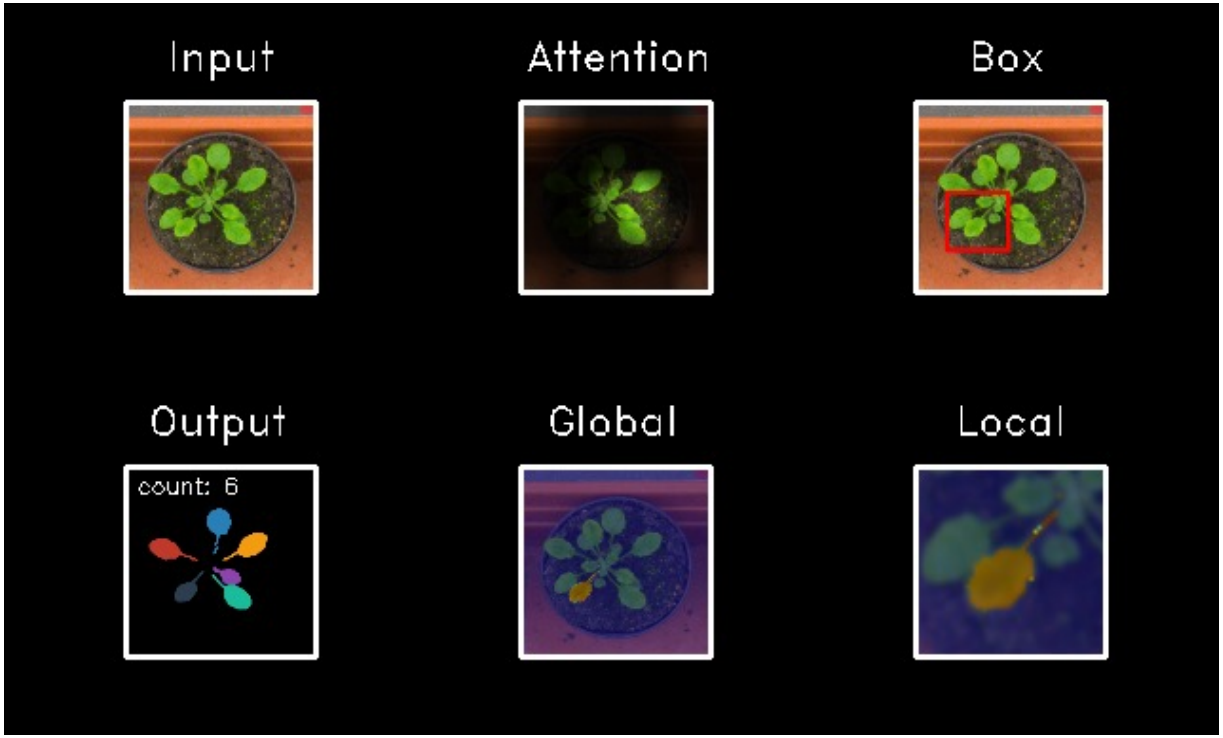


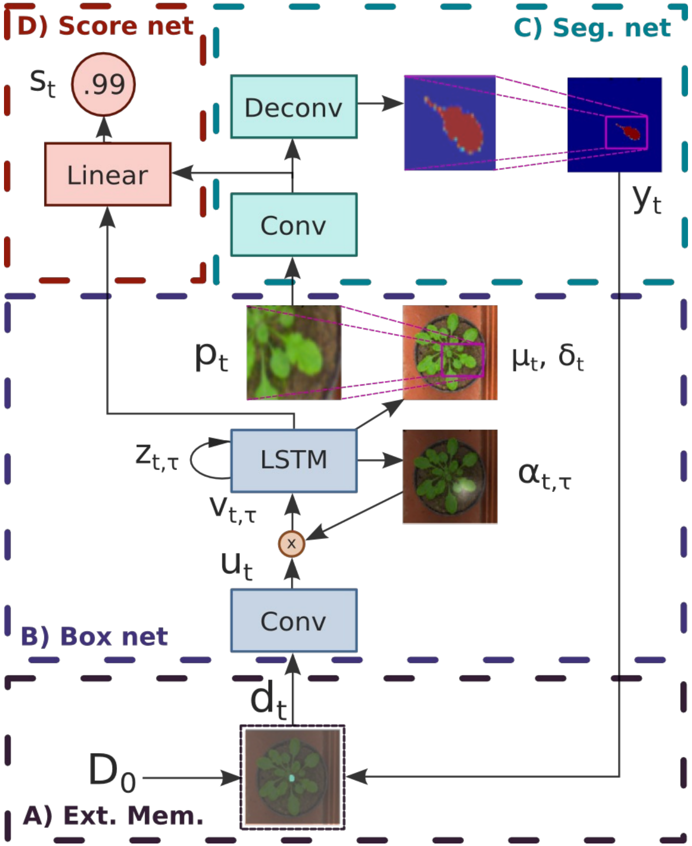


[Ren & Zemel, 2017]

NYU

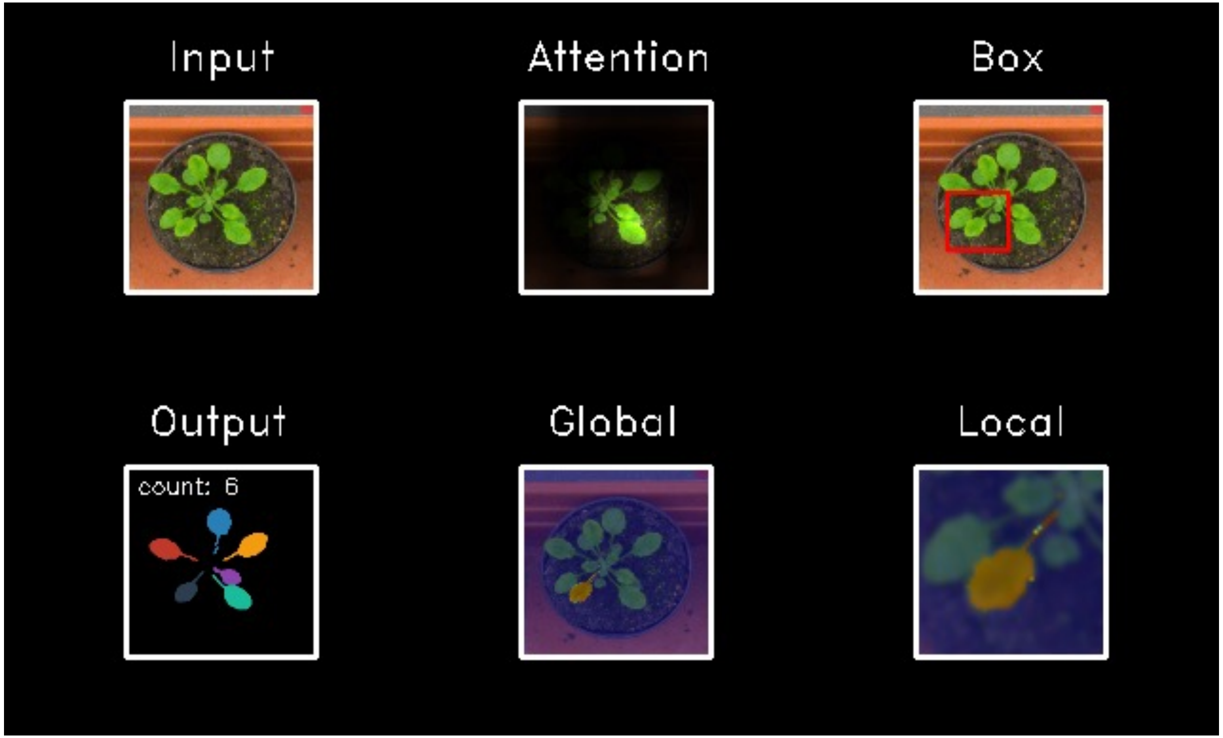
Demo

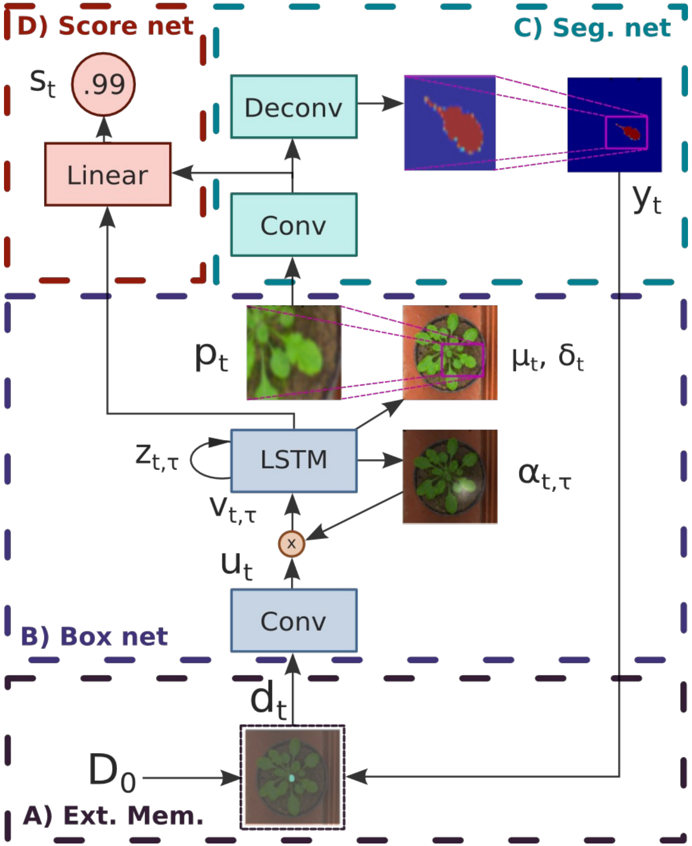




[Ren & Zemel, 2017]

Demo

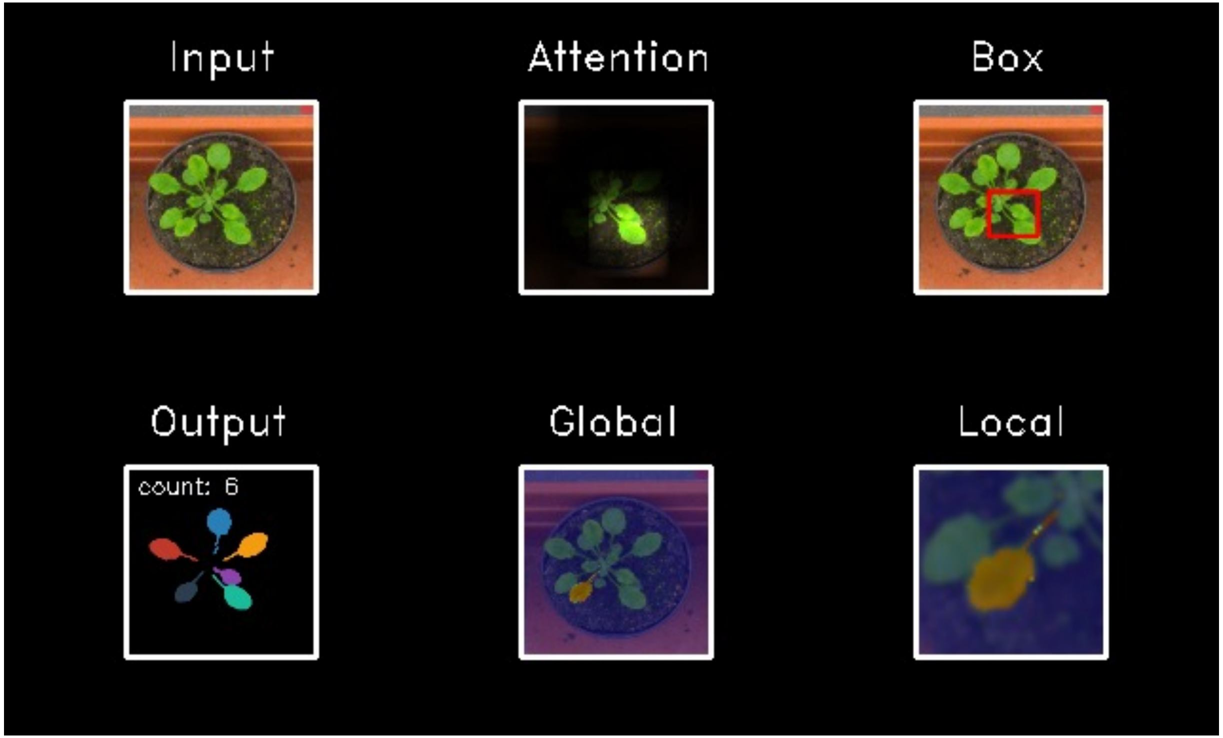


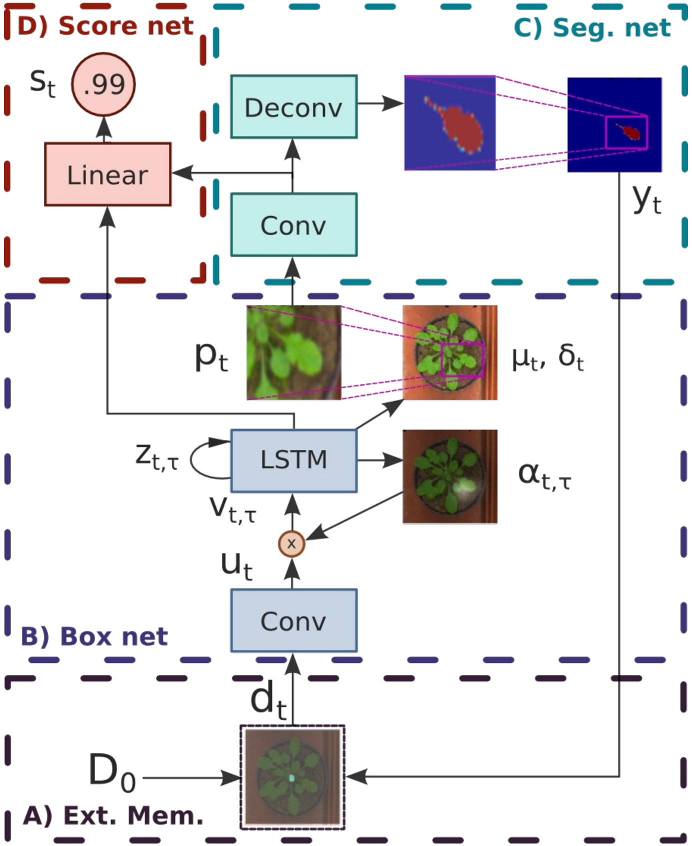


[Ren & Zemel, 2017]

NYU

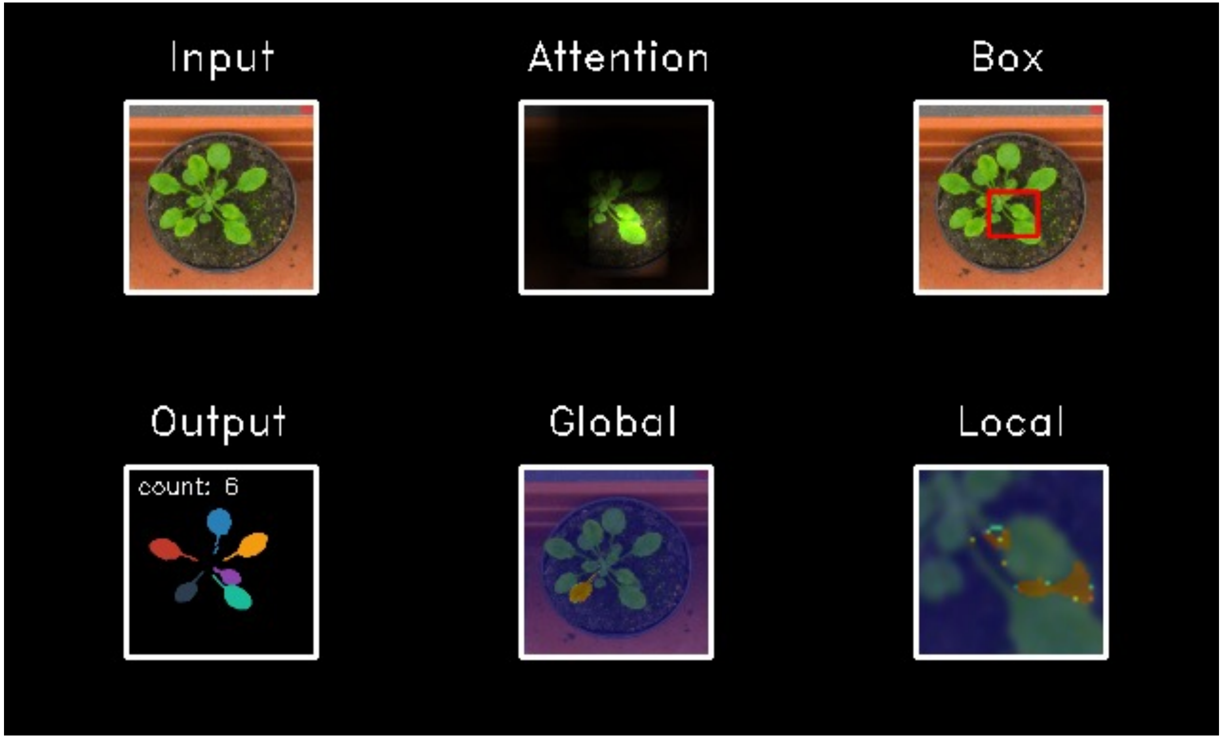
Demo

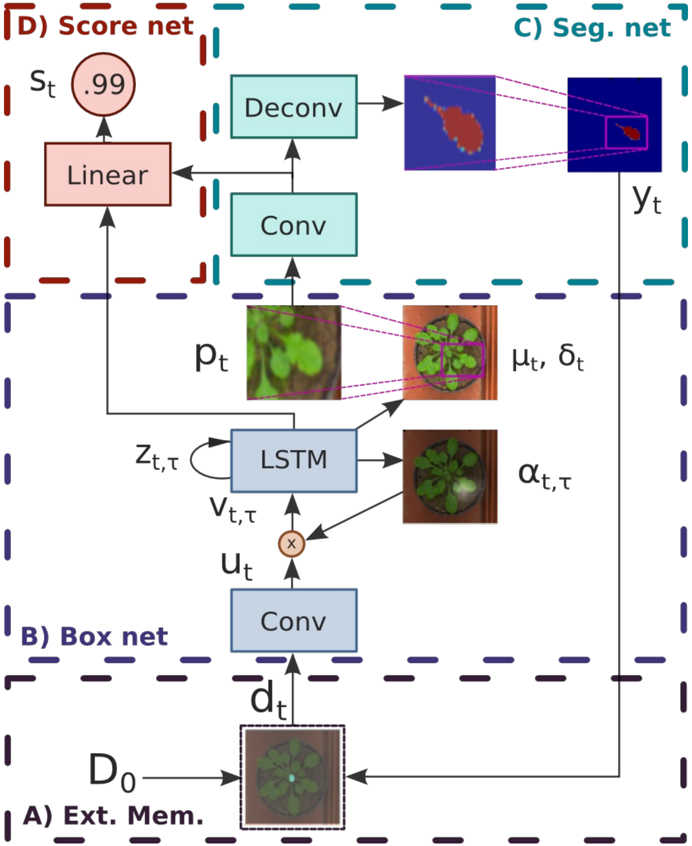




[Ren & Zemel, 2017]

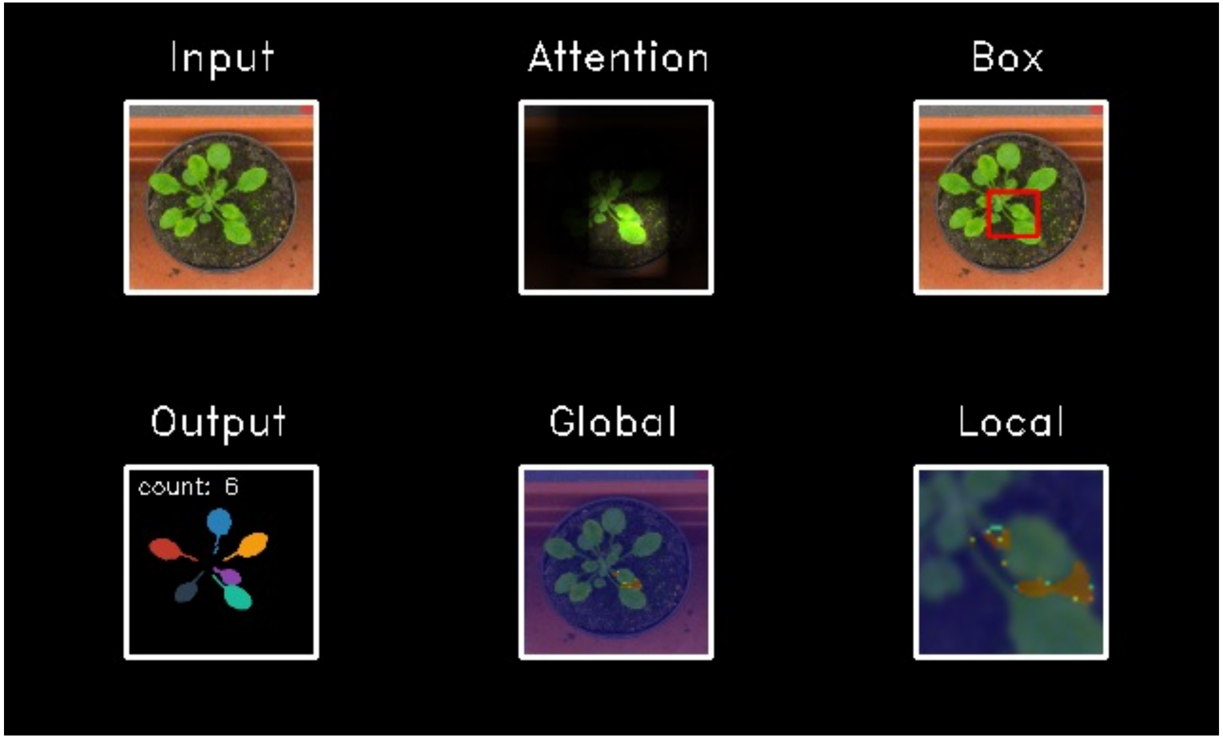
Demo

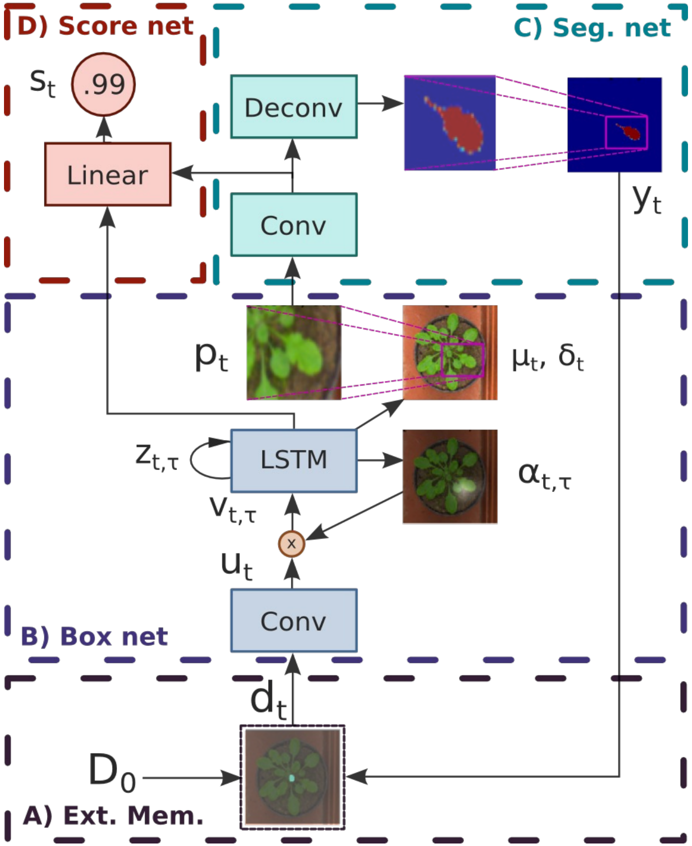




[Ren & Zemel, 2017]

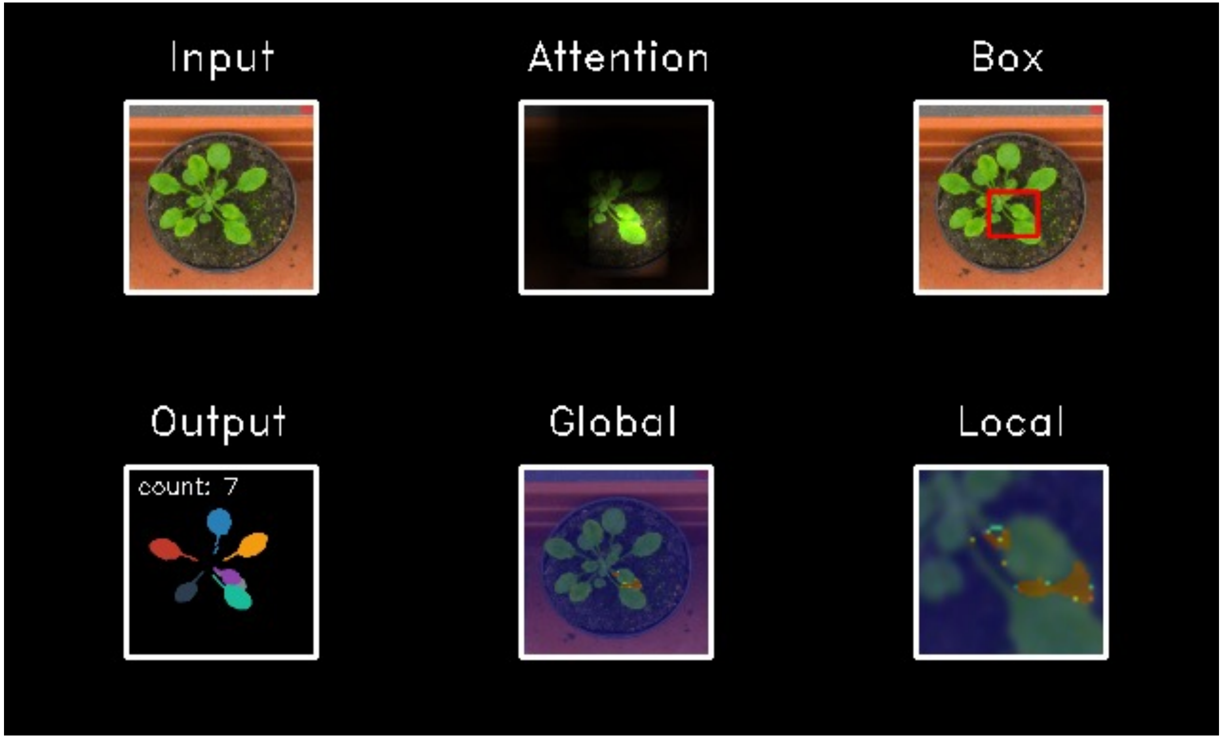
Demo

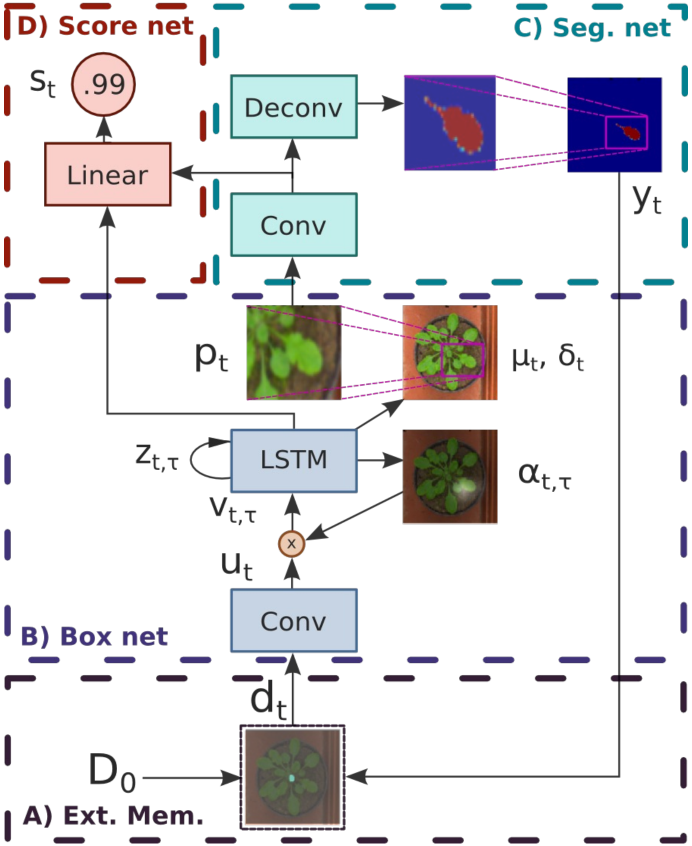




[Ren & Zemel, 2017]

Demo

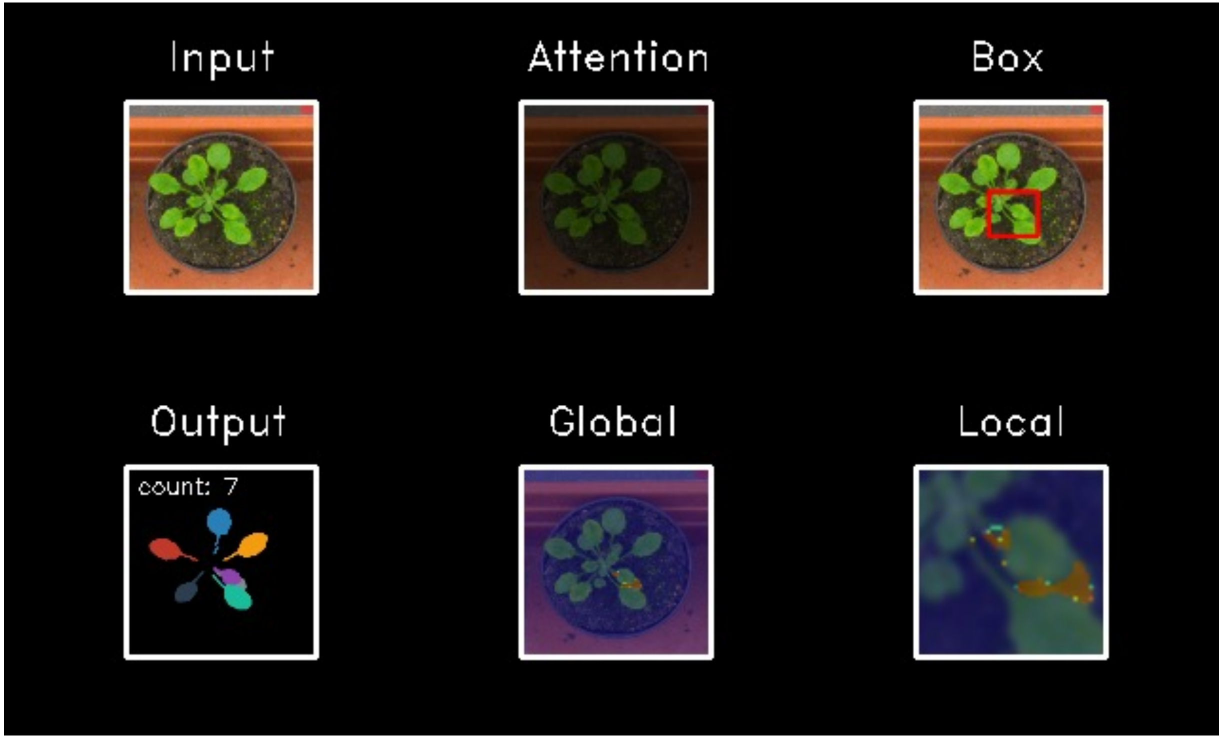


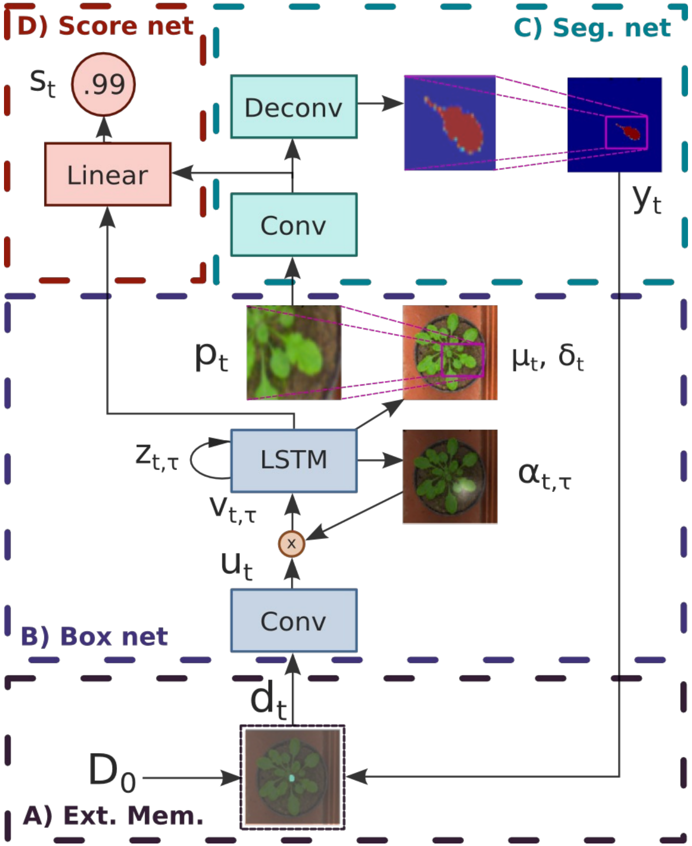


[Ren & Zemel, 2017]

NYU

Demo

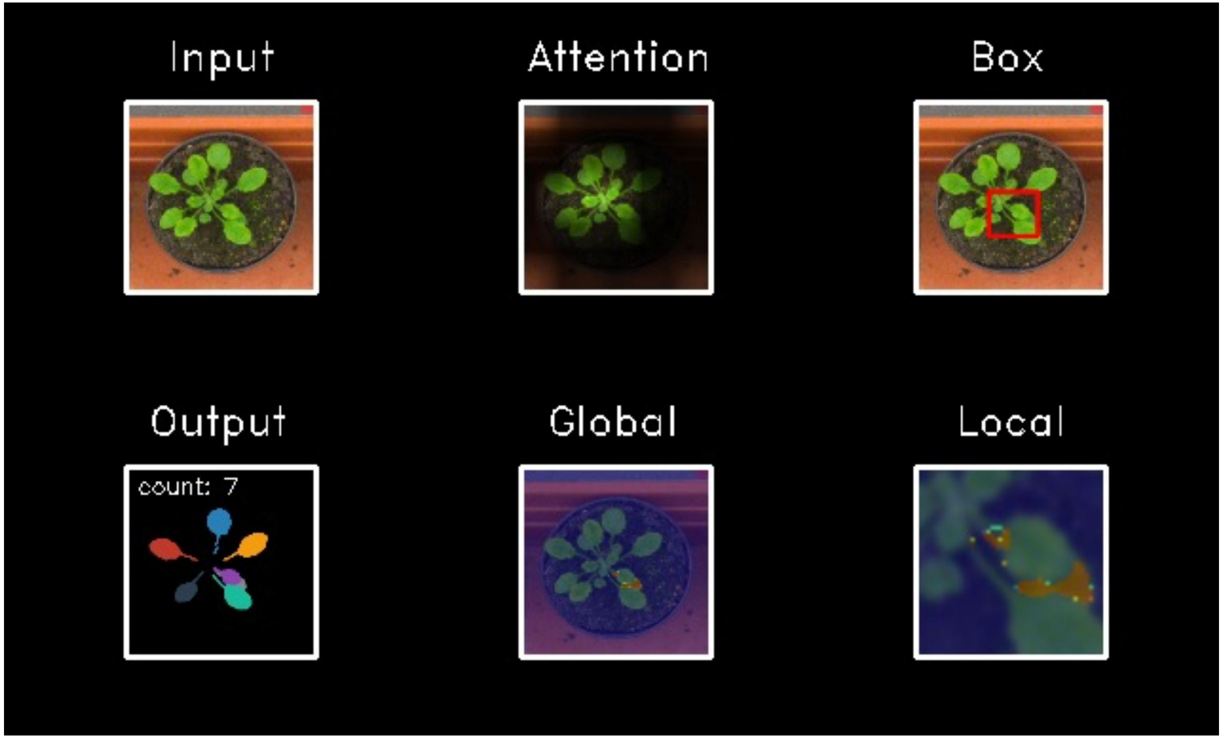


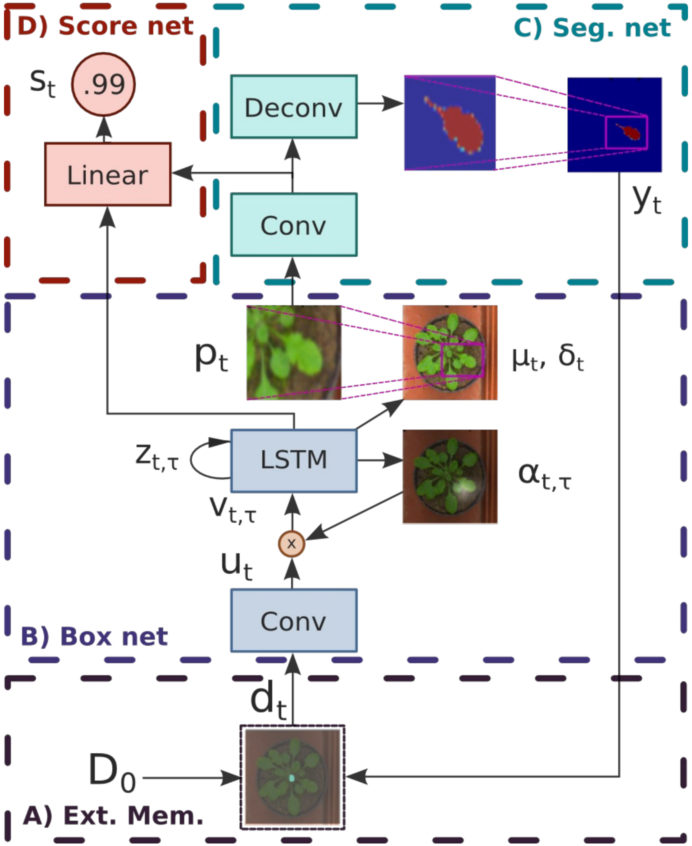


[Ren & Zemel, 2017]

NYU

Demo

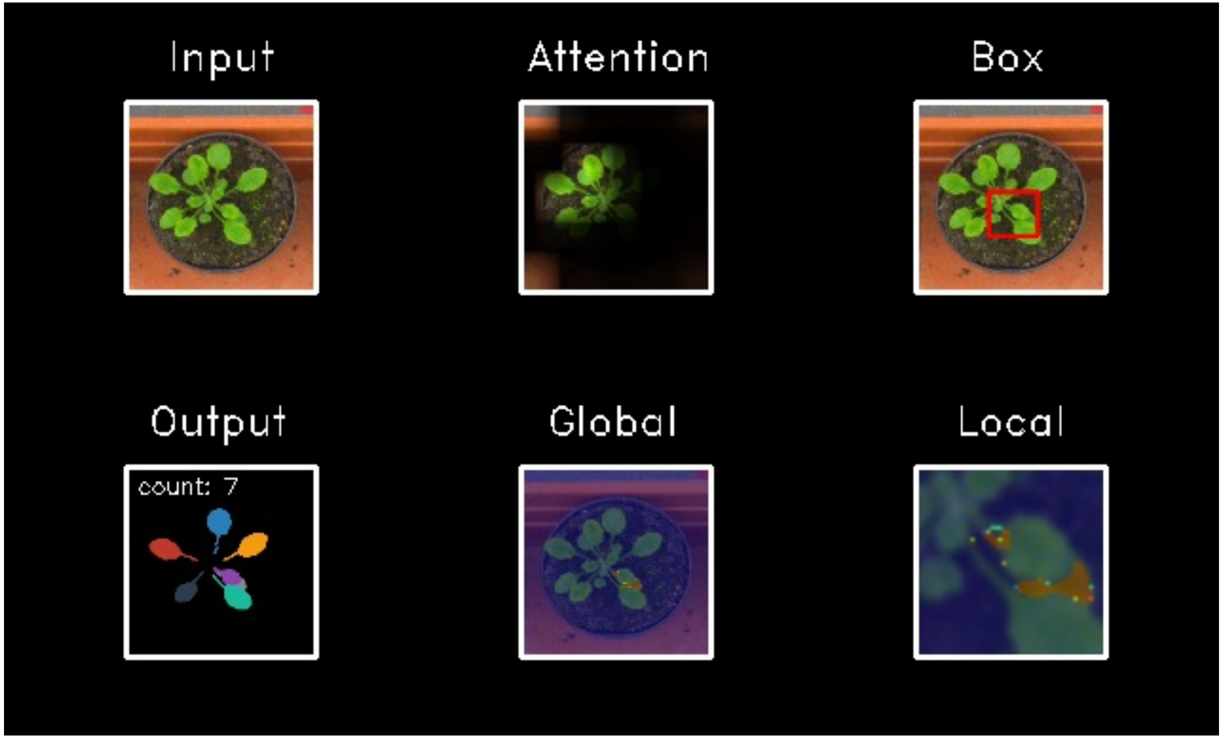


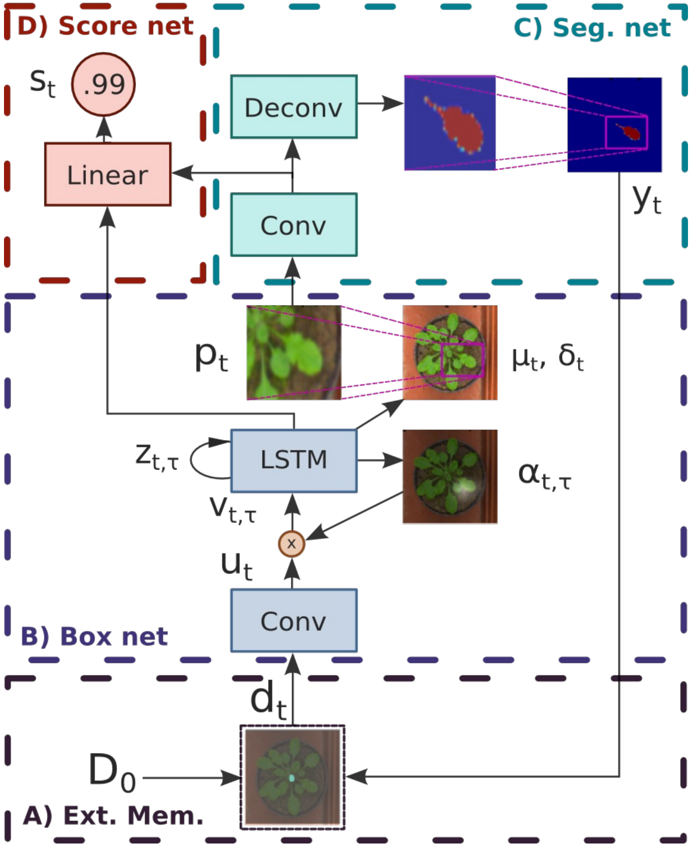


[Ren & Zemel, 2017]

NYU

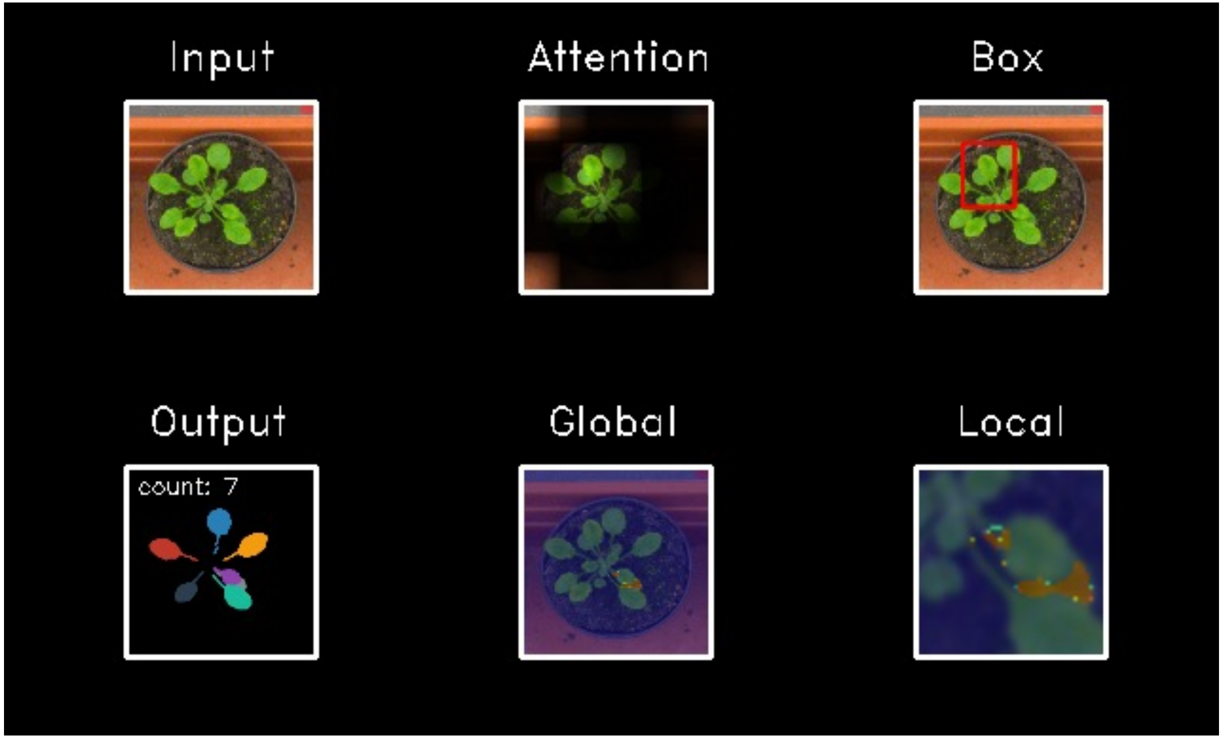
Demo

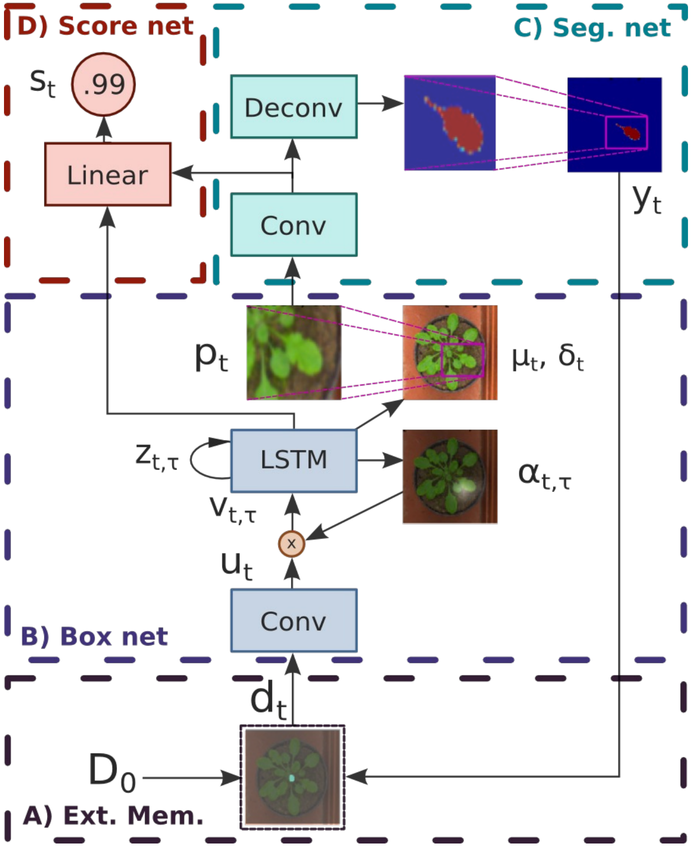




[Ren & Zemel, 2017]

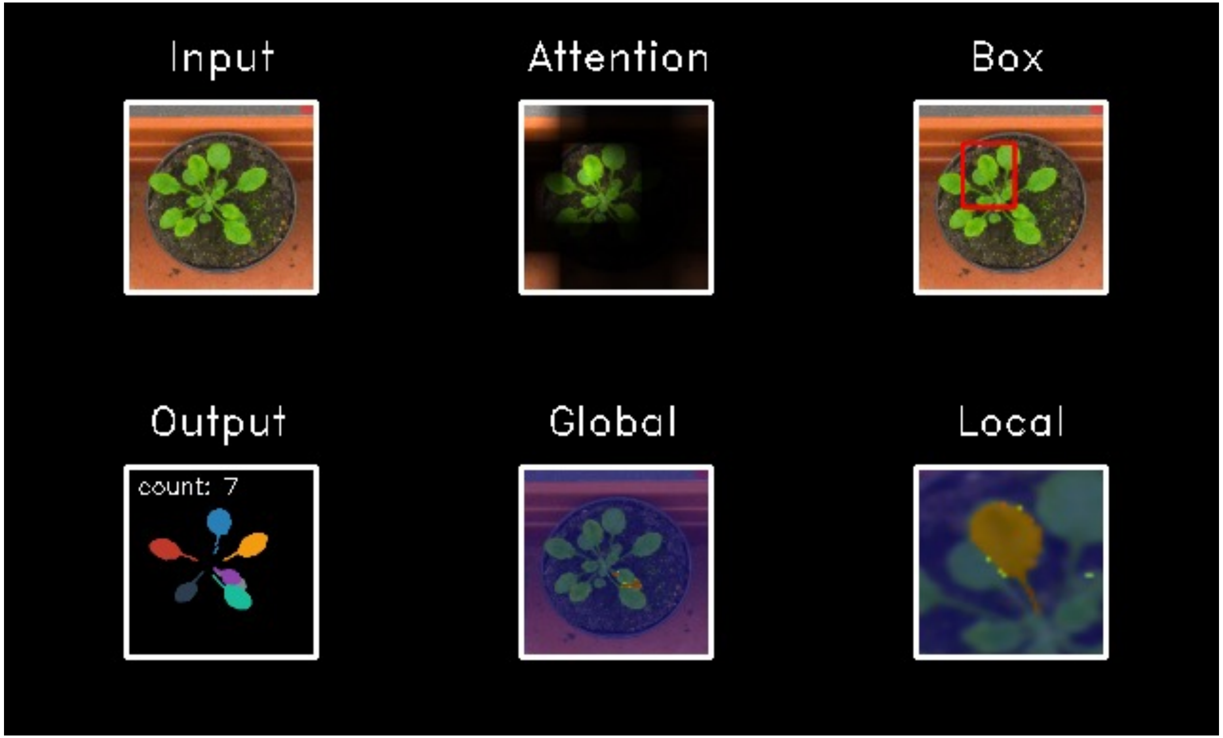
Demo

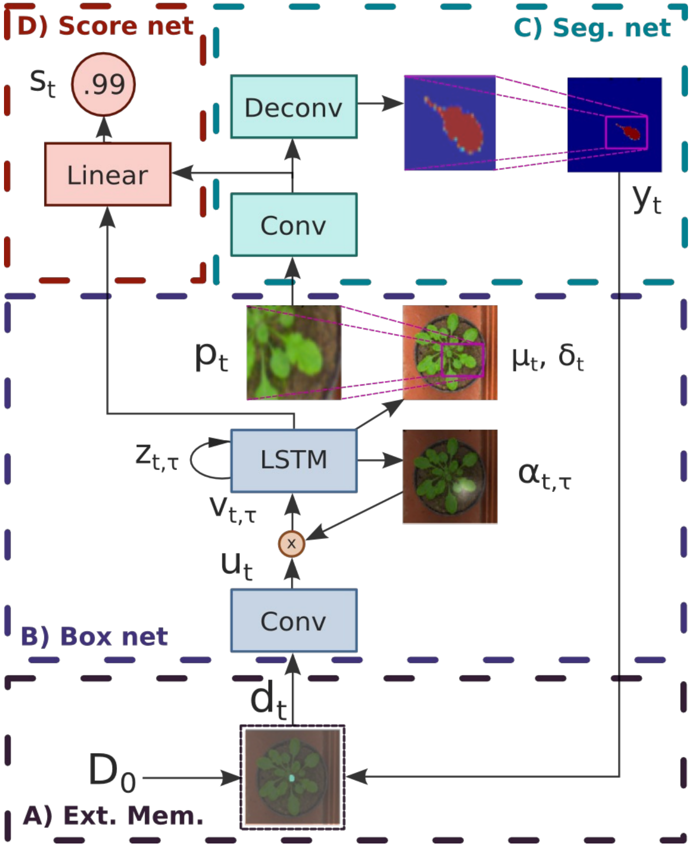




[Ren & Zemel, 2017]

Demo

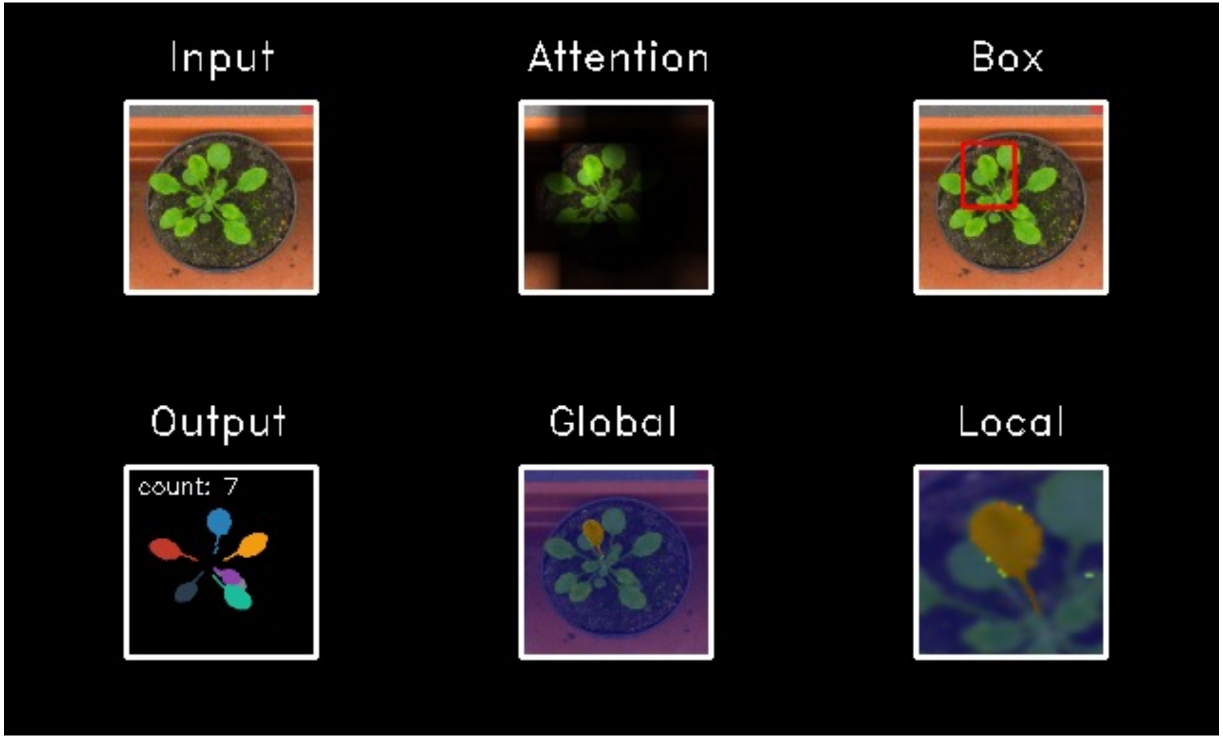


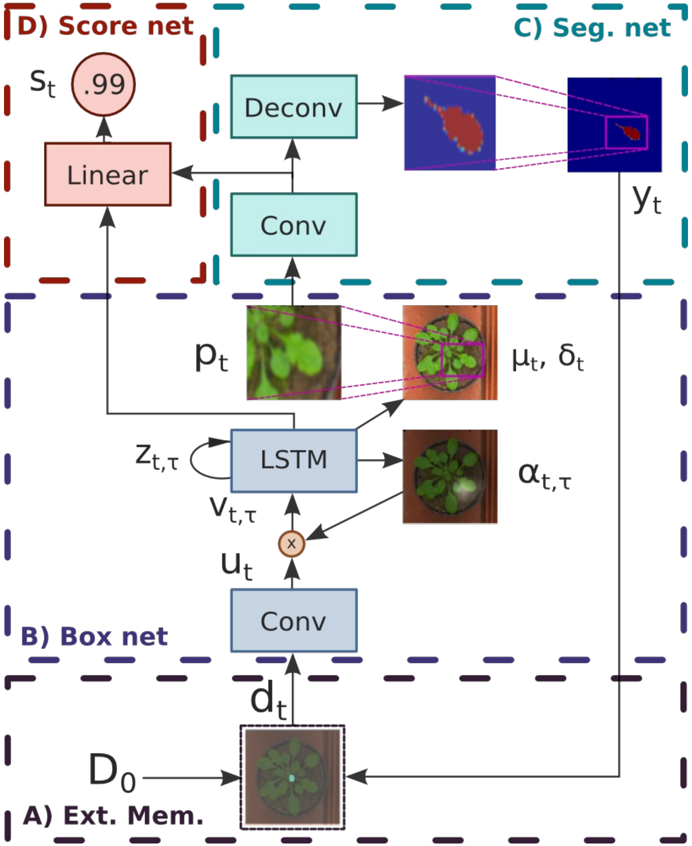


[Ren & Zemel, 2017]

NYU

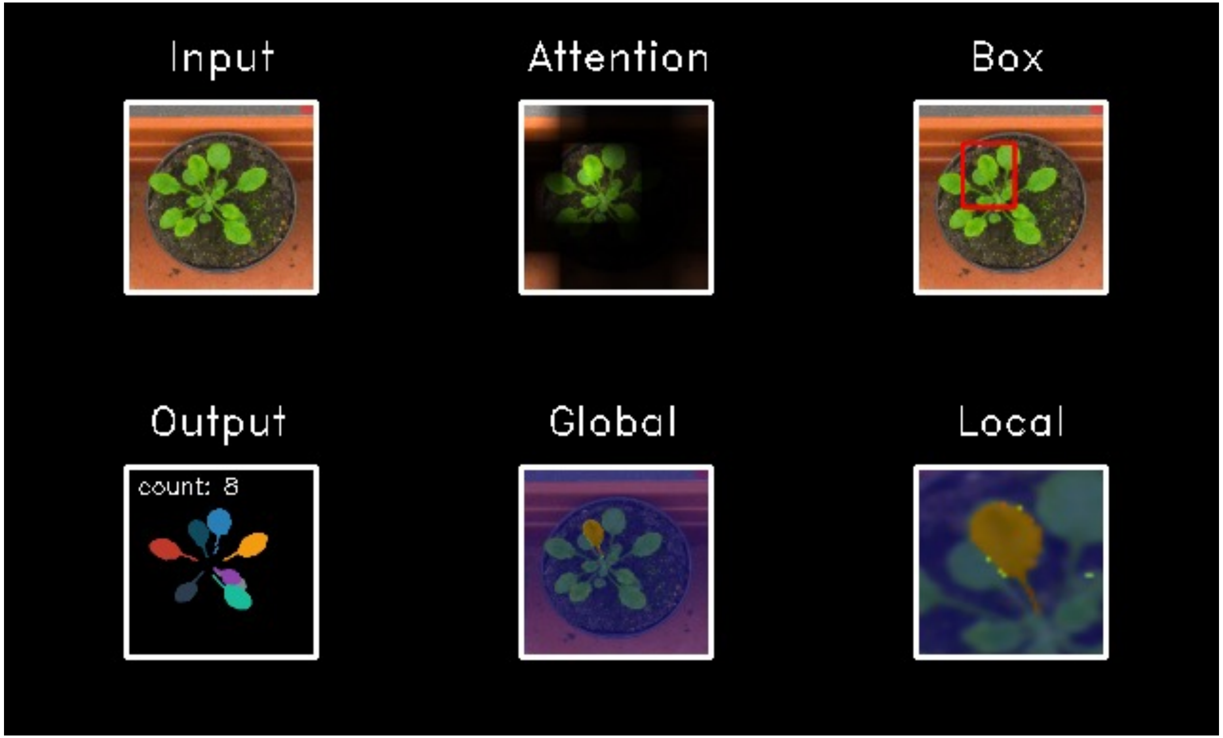
Demo

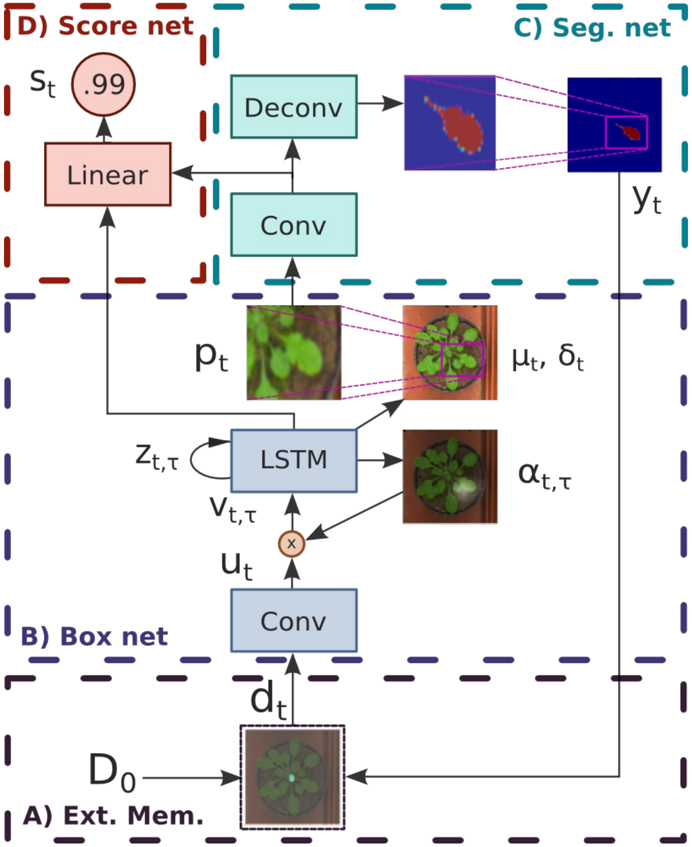




[Ren & Zemel, 2017]

Demo





[Ren & Zemel, 2017]

NYU

Summary

- Autoregressive models
  - Exact likelihood (instead of a lower bound or approximation)
  - Conditional probability may be easier to model
  - Need an ordering
  - Can take a long time to decode

Summary

- Autoregressive models
  - Exact likelihood (instead of a lower bound or approximation)
  - Conditional probability may be easier to model
  - Need an ordering
  - Can take a long time to decode
- Autoregressive models aren't just for language.
  - Planning
  - Set prediction problems
  - Perception

Summary

- Autoregressive models
  - Exact likelihood (instead of a lower bound or approximation)
  - Conditional probability may be easier to model
  - Need an ordering
  - Can take a long time to decode
- Autoregressive models aren't just for language.
  - Planning
  - Set prediction problems
  - Perception
- Generating the next variable is a different objective from learning a good representation of the whole sequence.

Summary

- Autoregressive models
  - Exact likelihood (instead of a lower bound or approximation)
  - Conditional probability may be easier to model
  - Need an ordering
  - Can take a long time to decode
- Autoregressive models aren’t just for language.
  - Planning
  - Set prediction problems
  - Perception
- Generating the next variable is a different objective from learning a good representation of the whole sequence.
- No hierarchical planning.

NYU

Energy-Based Modeling

- The goal is to learn a distribution $p(x)$ to model the set of examples.

Energy-Based Modeling

- The goal is to learn a distribution $p(x)$ to model the set of examples.
- Assuming Boltzmann distribution:

$$p(x;\theta) = \frac{1}{Z(\theta)} \exp(-E(x;\theta))$$

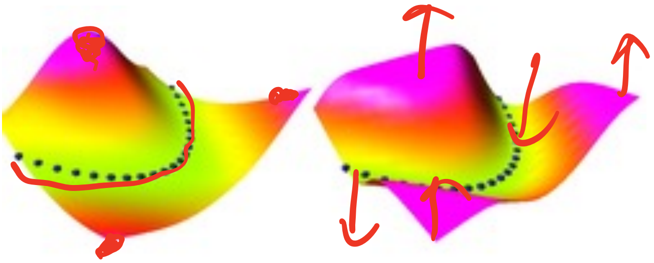
$$Z(\theta) = \int_X \exp(-E(x;\theta))$$

Energy-Based Modeling

- The goal is to learn a distribution $p(x)$ to model the set of examples.
- Assuming Boltzmann distribution:

$$p(x;\theta) = \frac{1}{Z(\theta)} \exp(-E(x;\theta))$$

$$Z(\theta) = \int_X \exp(-E(x;\theta))$$



Picture from LeCun

- Maximizing the likelihood:

↑ energy ↓ energy.

$$\frac{\partial}{\partial\theta} \log p(x;\theta) = \mathbb{E}_{x' \sim p(x)}\left[\frac{\partial E(x';\theta)}{\partial\theta}\right] - \frac{\partial E(x;\theta)}{\partial\theta}.$$

pos.

NYU

Structured Prediction

- EBM can be easily adapted to model the joint distribution of x and y.
- Requires optimization of y at inference time.

$$\arg\min_{y} E(x, y).$$

Optimization

- Computing $\mathbb{E}_{x' \sim p(x)} \left[\frac{\partial E(x';\theta)}{\partial \theta} \right]$ is non-trivial. We need to sample from p(x).

Optimization

- Computing $\mathbb{E}_{x' \sim p(x)} \left[\frac{\partial E(x';\theta)}{\partial \theta} \right]$ is non-trivial. We need to sample from p(x).
- But we only know $E(x)$!

$\frac{1}{Z} \exp(-E(x))$

Optimization

- Computing $\mathbb{E}_{x' \sim p(x)}\left[\frac{\partial E(x';\theta)}{\partial \theta}\right]$ is non-trivial. We need to sample from p(x).
- But we only know $E(x)$!
- Through MCMC samplers: MH, Langevin, HMC, Gibbs.

Optimization

- Computing $\mathbb{E}_{x' \sim p(x)} \left[\frac{\partial E(x';\theta)}{\partial \theta} \right]$ is non-trivial. We need to sample from p(x).
- But we only know $E(x)$!
- Through MCMC samplers: MH, Langevin, HMC, Gibbs.
- Approximations: Using truncated steps (Contrastive Divergence)

Optimization

- Computing $\mathbb{E}_{x' \sim p(x)}\left[\frac{\partial E(x';\theta)}{\partial \theta}\right]$ is non-trivial. We need to sample from p(x).
- But we only know $E(x)$!
- Through MCMC samplers: MH, Langevin, HMC, Gibbs.
- Approximations: Using truncated steps (Contrastive Divergence)
- Score Matching: Tries to model $\nabla_x \log p_{\mathrm{data}}(x)$ and $\nabla_x \log p_\theta(x)$
  - If we know the gradient, we can improve the samples.
  - Closely related to diffusion models

NYU

Max-Entropy Inverse RL

- Inverse RL for learning the reward function.

Ziebart et al. Maximum Entropy Inverse Reinforcement Learning. AAAI 2008.

Max-Entropy Inverse RL

- Inverse RL for learning the reward function.
- In practice, use rewards as negative energy.

Ziebart et al. Maximum Entropy Inverse Reinforcement Learning. AAAI 2008.

Max-Entropy Inverse RL

- Inverse RL for learning the reward function.
- In practice, use rewards as negative energy.

$$\max_{\psi} \frac{1}{N}\sum_{i=1}^{N} r_\psi(\tau_i) - \log Z \qquad Z = \int p(\tau)\exp(r_\psi(\tau))d\tau$$

$$\nabla_\psi \mathcal{L} = \frac{1}{N}\sum_{i=1}^{N} \nabla_\psi r_\psi(\tau_i) - \underbrace{\frac{1}{Z}\int p(\tau)\exp(r_\psi(\tau))}_{p(\tau|\mathcal{O}_{1:T},\psi)}\nabla_\psi r_\psi(\tau)d\tau$$

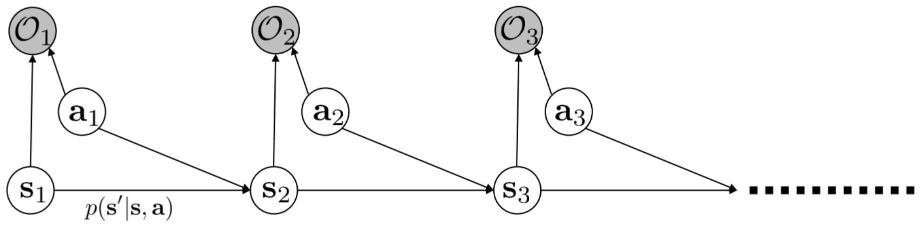
$$\nabla_\psi \mathcal{L} = \underbrace{E_{\tau\sim\pi^\star(\tau)}[\nabla_\psi r_\psi(\tau_i)]}_{\text{estimate with expert samples}} - \underbrace{E_{\tau\sim p(\tau|\mathcal{O}_{1:T},\psi)}}_{\text{soft optimal policy under current reward}}[\nabla_\psi r_\psi(\tau)]$$

Slide credit: Sergey Levine

Ziebart et al. Maximum Entropy Inverse Reinforcement Learning. AAAI 2008.

Max-Entropy Inverse RL

- In a discrete MDP, we can use DP to compute probability.



Slide credit: Sergey Levine

Max-Entropy Inverse RL

- In a discrete MDP, we can use DP to compute probability.

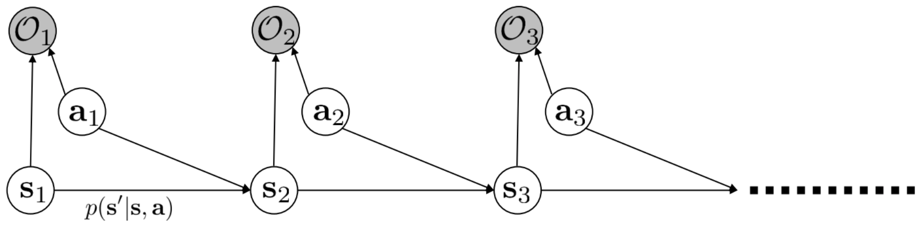
$$\nabla_\psi \mathcal{L} = E_{\tau\sim\pi^\star(\tau)}[\nabla_\psi r_\psi(\tau_i)] - \underbrace{E_{\tau\sim p(\tau|\mathcal{O}_{1:T},\psi)}[\nabla_\psi r_\psi(\tau)]}$$

let $\mu_t(\mathbf{s}_t, \mathbf{a}_t) \propto \beta(\mathbf{s}_t, \mathbf{a}_t)\alpha(\mathbf{s}_t)$

$$\sum_{t=1}^{T} \int \int \mu_t(\mathbf{s}_t, \mathbf{a}_t) \nabla_\psi r_\psi(\mathbf{s}_t, \mathbf{a}_t) d\mathbf{s}_t d\mathbf{a}_t$$

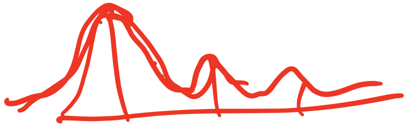
$$= \sum_{t=1}^{T} \vec{\mu}_t^T \nabla_\psi \vec{r}_\psi$$

state-action visitation probability for each $(\mathbf{s}_t, \mathbf{a}_t)$



Slide credit: Sergey Levine

Max-Entropy Inverse RL



- In a discrete MDP, we can use DP to compute probability.

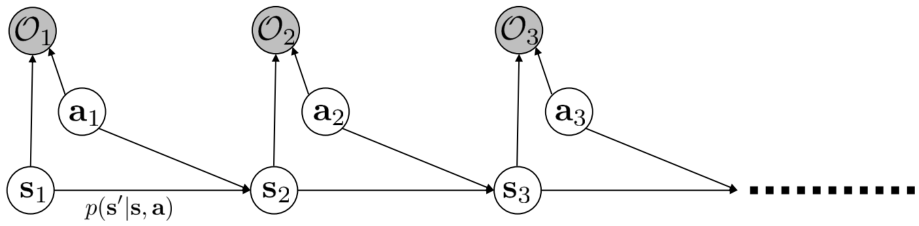
$$\nabla_\psi \mathcal{L} = E_{\tau \sim \pi^\star(\tau)}[\nabla_\psi r_\psi(\tau_i)] - \underbrace{E_{\tau \sim p(\tau|\mathcal{O}_{1:T},\psi)}[\nabla_\psi r_\psi(\tau)]}$$

$$\sum_{t=1}^{T} \int \int \mu_t(\mathbf{s}_t, \mathbf{a}_t) \nabla_\psi r_\psi(\mathbf{s}_t, \mathbf{a}_t) d\mathbf{s}_t d\mathbf{a}_t$$

$$= \sum_{t=1}^{T} \vec{\mu}_t^T \nabla_\psi \vec{r}_\psi$$

state-action visitation probability for each $(\mathbf{s}_t, \mathbf{a}_t)$

let $\mu_t(\mathbf{s}_t, \mathbf{a}_t) \propto \beta(\mathbf{s}_t, \mathbf{a}_t)\alpha(\mathbf{s}_t)$



Slide credit: Sergey Levine

in the case where $r_\psi(\mathbf{s}_t, \mathbf{a}_t) = \psi^T \mathbf{f}(\mathbf{s}_t, \mathbf{a}_t)$, we can show that it optimizes

$$\max_\psi \mathcal{H}(\pi^{r_\psi}) \text{ such that } E_{\pi^{r_\psi}}[\mathbf{f}] = E_{\pi^\star}[\mathbf{f}]$$

optimal max-ent policy under r^ψ

unknown expert policy estimated with samples

as random as possible while matching features

Max-Margin Learning

SSVM

CRF

- In addition to probabilistic learning, we can also apply the max-margin framework.

Max-Margin Learning

- In addition to probabilistic learning, we can also apply the max-margin framework.
- Review: SSVM vs. CRF.

Max-Margin Learning

- In addition to probabilistic learning, we can also apply the max-margin framework.
- Review: SSVM vs. CRF.

x expert

$$\arg\min_\theta \sum_{n=1}^{N} \max[0, m + E(x_n;\theta) - E(x^*;\theta)] + \lambda||\theta||_2^2.$$

NYU

Max-Margin Learning

- In addition to probabilistic learning, we can also apply the max-margin framework.
- Review: SSVM vs. CRF.

$$\arg\min_\theta \sum_{n=1}^{N} \max[0, m + E(x_n; \theta) - E(x^*; \theta)] + \lambda||\theta||_2^2.$$

planning.

- Margin can be difference in trajectories.

NYU

Max-Margin Learning

- In addition to probabilistic learning, we can also apply the max-margin framework.
- Review: SSVM vs. CRF.

$$\arg\min_\theta \sum_{n=1}^{N} \max[0, m + E(x_n;\theta) - E(x^*;\theta)] + \lambda||\theta||_2^2.$$

- Margin can be difference in trajectories.
- Non-probabilistic

NYU

Max-Margin Learning

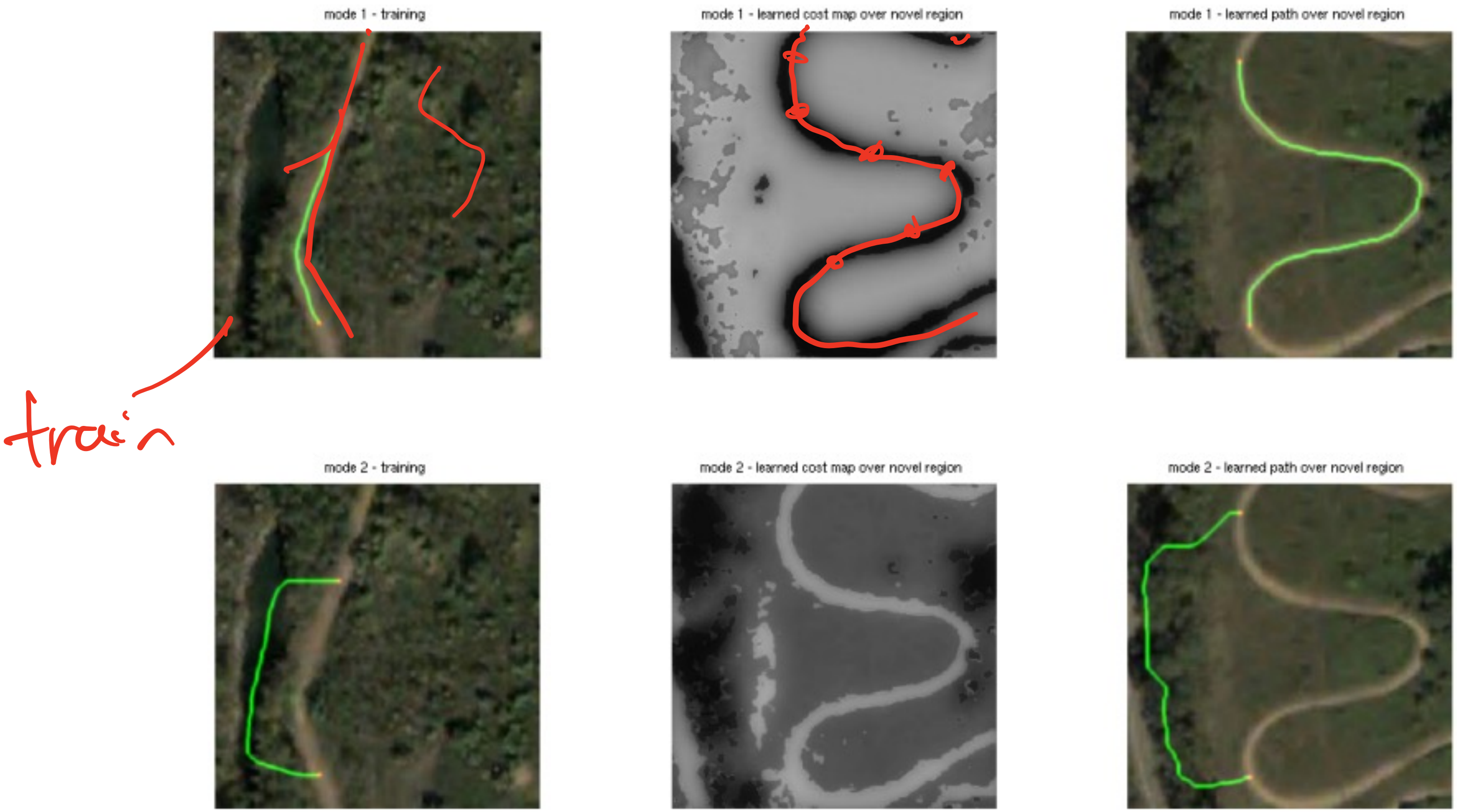
90%

- In addition to probabilistic learning, we can also apply the max-margin framework.
- Review: SSVM vs. CRF.

$$\arg\min_{\theta} \sum_{n=1}^{N} \max[0, m + E(x_n;\theta) - E(x^*;\theta)] + \lambda||\theta||_2^2.$$

- Margin can be difference in trajectories.
- Non-probabilistic
- Still need to run optimization to find the best x^*

Max-Margin Planning



Ratliff et al. Maximum Margin Planning. ICML 2006.

Diffusion Models

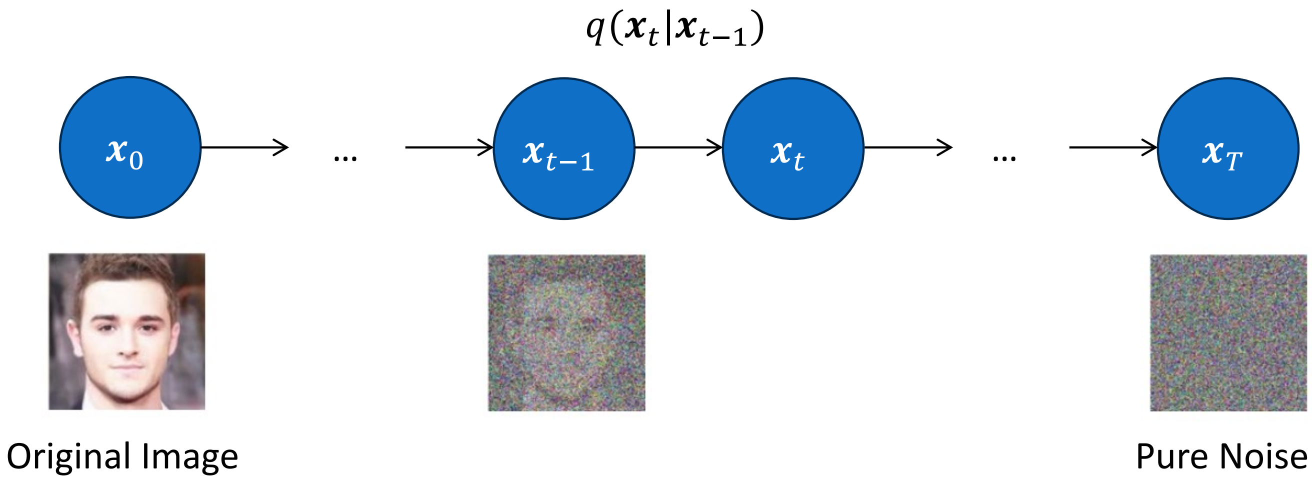
- A popular model for generative model today is diffusion model.

Diffusion Models

- A popular model for generative model today is diffusion model.
- The intuition is to iteratively denoise from Gaussian random noises into an image.

Diffusion Models

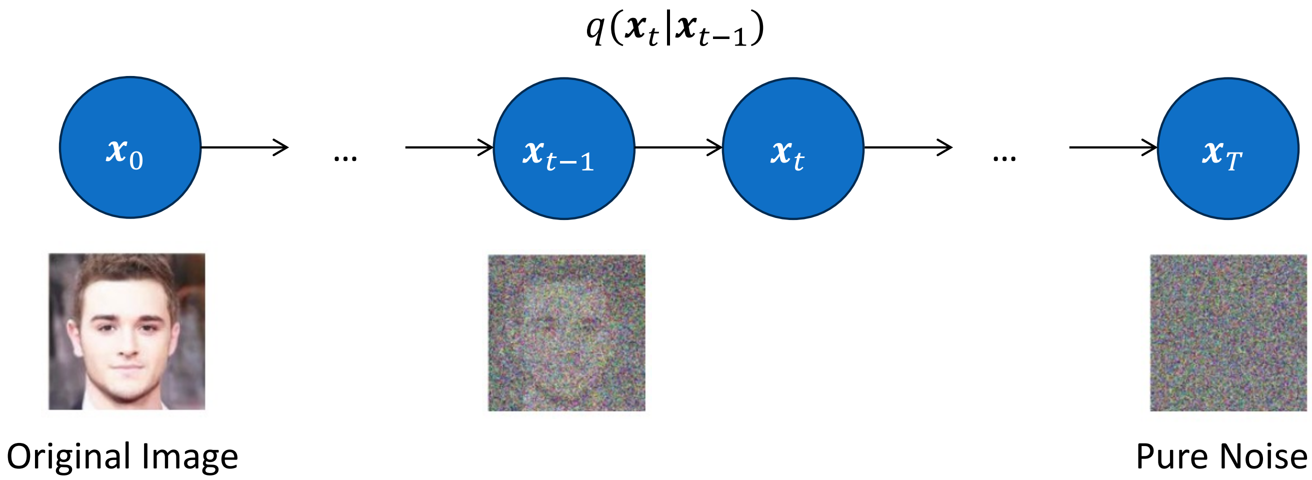
- A popular model for generative model today is diffusion model.
- The intuition is to iteratively denoise from Gaussian random noises into an image.



NYU

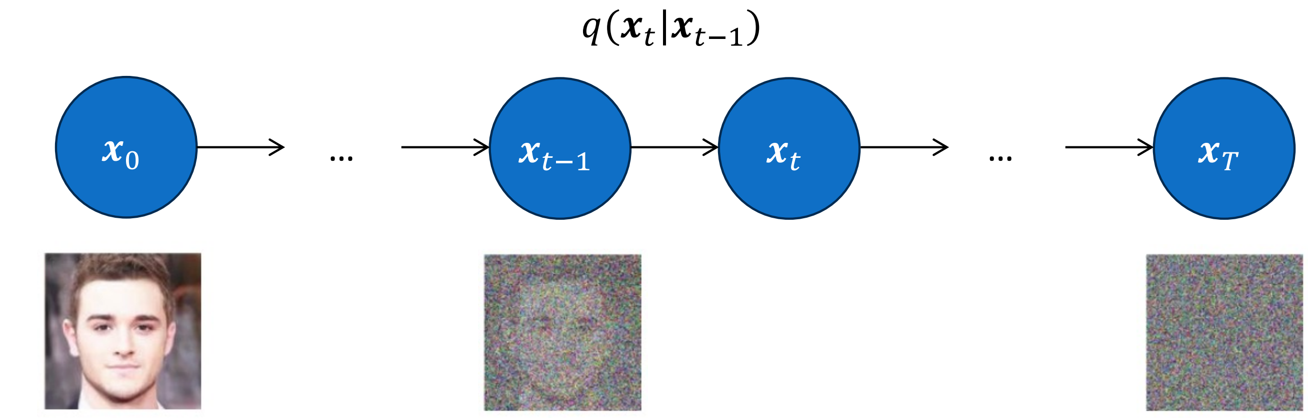
Diffusion Models

- Forward process: $q(x_t|x_{t-1}) = \mathcal{N}(x_t; \sqrt{1-\beta_t}x_{t-1}, \beta_t I).$



Diffusion Models

- Forward process: $q(x_t|x_{t-1}) = \mathcal{N}(x_t; \sqrt{1-\beta_t}x_{t-1}, \beta_t I)$.
- You can also write: $x_t = \sqrt{1-\beta_t}x_{t-1} + \sqrt{\beta_t}\epsilon_t,\ \epsilon_t \sim \mathcal{N}(0, I)$.



Original Image

Pure Noise

Properties of the Forward Process

- Forward process: $x_t = \sqrt{1-\beta_t}x_{t-1} + \sqrt{\beta_t}\epsilon_t$

Properties of the Forward Process

- Forward process: $x_t = \sqrt{1 - \beta_t} x_{t-1} + \sqrt{\beta_t} \epsilon_t$
- Write x_t as a function of x_0 with larger noises:

$$q(x_t | x_0) = \mathcal{N}(x_t; \sqrt{\bar{\alpha}_t} x_0, (1 - \bar{\alpha}_t) I).$$

Properties of the Forward Process

- Forward process: $x_t = \sqrt{1-\beta_t} x_{t-1} + \sqrt{\beta_t} \epsilon_t$
- Write x_t as a function of x_0 with larger noises:

$$q(x_t|x_0) = \mathcal{N}(x_t; \sqrt{\bar{\alpha}_t} x_0, (1-\bar{\alpha}_t) I).$$

- Cumulative schedule: $\alpha_t = 1 - \beta_t$. $\bar{\alpha}_t = \prod_{s=1}^{t} \alpha_s$.

Properties of the Forward Process

- Forward process: $x_t = \sqrt{1-\beta_t}x_{t-1} + \sqrt{\beta_t}\epsilon_t$
- Write x_t as a function of x_0 with larger noises:

$$q(x_t|x_0) = \mathcal{N}(x_t; \sqrt{\bar{\alpha}_t}x_0, (1-\bar{\alpha}_t)I).$$

- Cumulative schedule: $\alpha_t = 1 - \beta_t$. $\bar{\alpha}_t = \prod_{s=1}^{t} \alpha_s$.
- $x_t = \sqrt{\bar{\alpha}_t}x_0 + \sqrt{1-\bar{\alpha}_t}\epsilon$.

NYU

Cumulative Schedule

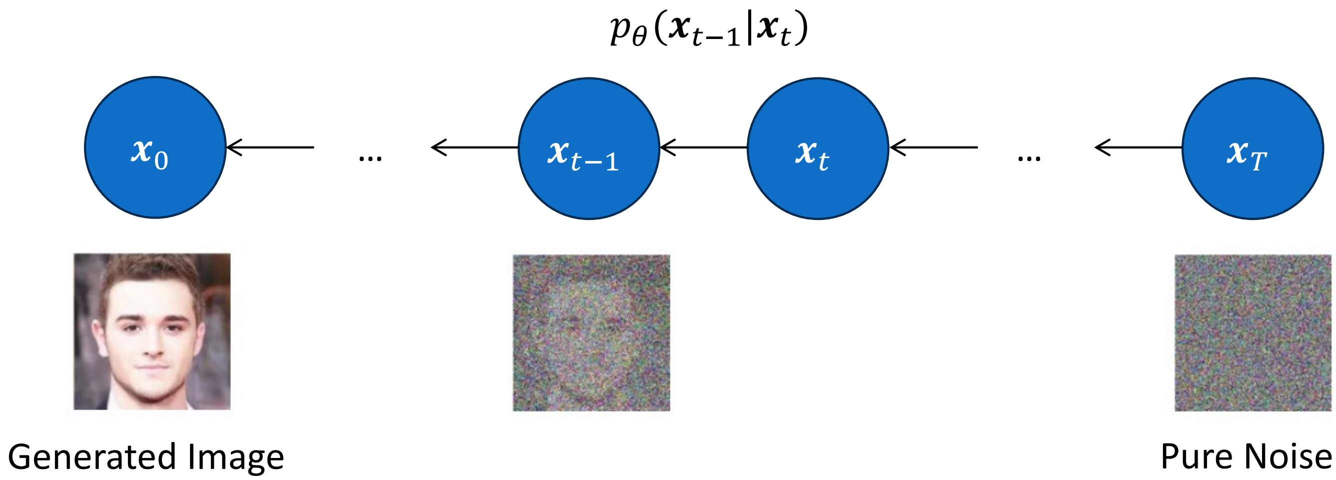
$$\alpha_t = 1 - \beta_t.$$

- Show it's true for x_2: $\bar{\alpha}_t = \prod_{s=1}^{t} \alpha_s.$

$$\begin{aligned} x_2 &= \sqrt{1-\beta_2}x_1 + \sqrt{\beta_2}\epsilon_2 = \sqrt{1-\beta_2}\sqrt{1-\beta_1}x_0 + \sqrt{\beta_2}\epsilon_2 + \sqrt{1-\beta_2}\sqrt{\beta_1}\epsilon_1 \\ &= \alpha_1\alpha_2 x_0 + \sqrt{(1-\beta_2)\beta_1 + \beta_2}\epsilon \\ &= \bar{\alpha}_2 x_0 + \sqrt{1-(1-\beta_1)(1-\beta_2)}\epsilon \\ &= \bar{\alpha}_2 x_0 + \sqrt{1-\bar{\alpha}_2}\epsilon. \end{aligned}$$

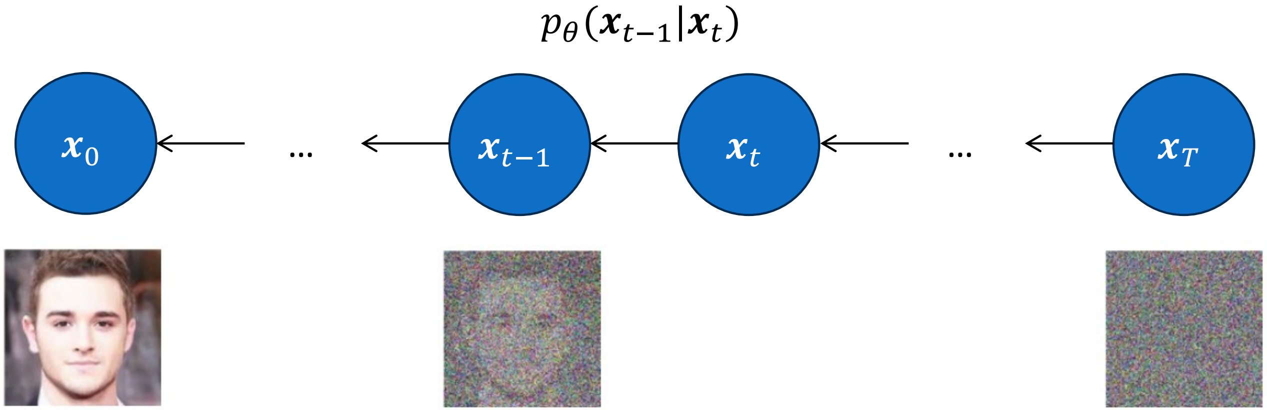
Reverse Process

- A generative model wants to predict x_0 from x_T.



Reverse Process

- A generative model wants to predict x_0 from x_T.
- The reverse process transition is also Gaussian distributed. But we don't know what the transition will be like just by looking at the noisy image!



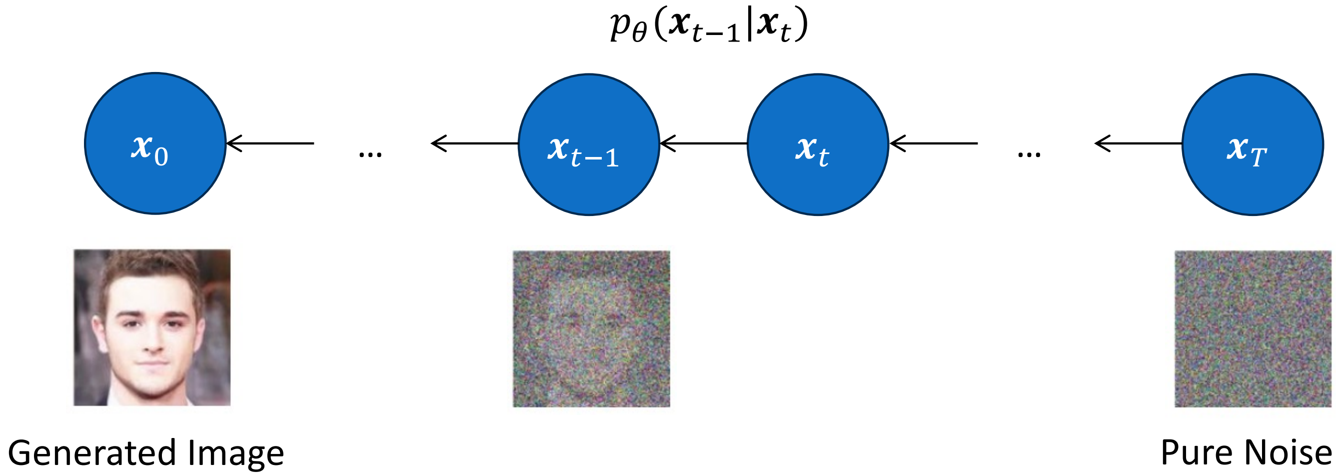
Generated Image

Pure Noise

Reverse Process

- So, we need to learn a "model":

$$p_\theta(x_{t-1}|x_t) = \mathcal{N}(x_{t-1}; \mu_\theta(x_t, t), \Sigma_\theta(x_t, t)).$$

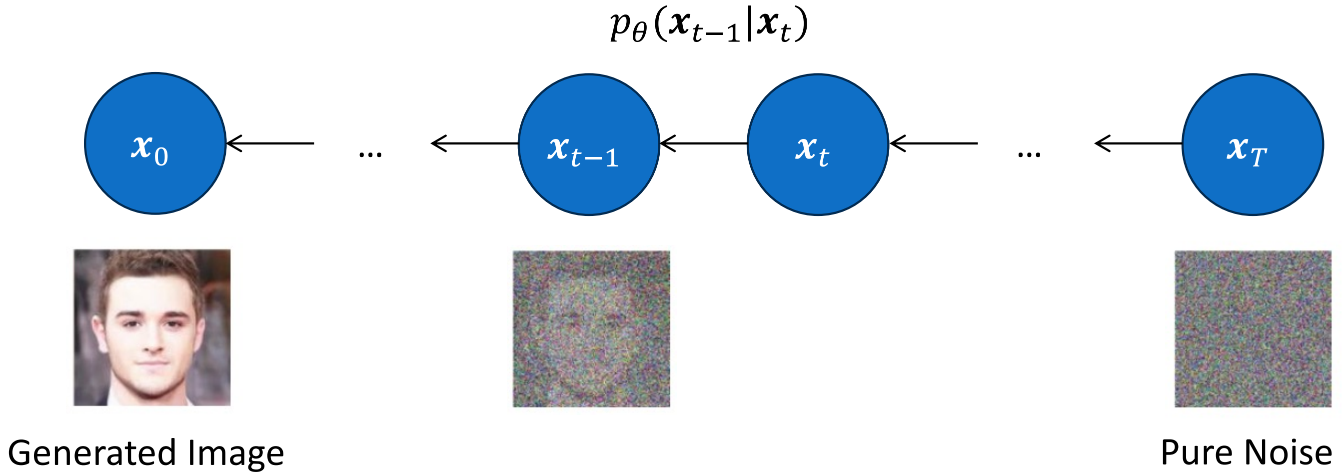


Reverse Process

- So, we need to learn a "model":

$$p_\theta(x_{t-1}|x_t) = \mathcal{N}(x_{t-1}; \mu_\theta(x_t, t), \Sigma_\theta(x_t, t)).$$

- μ_θ is the denoising vector.



NYU

Reverse Process

- Compute μ_θ? Derive $p(x_{t-1}|x_t)$.

Reverse Process

- Compute μ_θ? Derive $p(x_{t-1}|x_t)$.
- Bayes rule:

$$q(x_{t-1}|x_t) = \frac{q(x_t|x_{t-1})q(x_{t-1})}{q(x_t)}.$$

NYU

Reverse Process

- Compute μ_θ? Derive $p(x_{t-1}|x_t)$.
- Bayes rule:
$$q(x_{t-1}|x_t) = \frac{q(x_t|x_{t-1})q(x_{t-1})}{q(x_t)}.$$
- But we don't know the marginal distribution $q(x_{t-1})$. We only know q_T and $q(x_t|x_{t-1})$.

NYU

Reverse Process

- Compute μ_θ? Derive $p(x_{t-1}|x_t)$.
- Bayes rule:

$$q(x_{t-1}|x_t) = \frac{q(x_t|x_{t-1})q(x_{t-1})}{q(x_t)}.$$

- But we don't know the marginal distribution $q(x_{t-1})$. We only know q_T and $q(x_t|x_{t-1})$.
- Solution: Condition on the original input x_0:

$$q(x_{t-1}|x_t, x_0) = \frac{q(x_t|x_{t-1})q(x_{t-1}|x_0)}{q(x_t|x_0)}.$$

NYU

Reverse Process

$$q(x_{t-1}|x_t, x_0) = \frac{q(x_t|x_{t-1})q(x_{t-1}|x_0)}{q(x_t|x_0)}.$$

Reverse Process

$$q(x_{t-1}|x_t, x_0) = \frac{q(x_t|x_{t-1})q(x_{t-1}|x_0)}{q(x_t|x_0)}.$$

$$q(x_{t-1}|x_t, x_0) = \mathcal{N}(x_{t-1}; \tilde{\mu}_t, \tilde{\beta} I).$$

NYU

Reverse Process

$$q(x_{t-1}|x_t, x_0) = \frac{q(x_t|x_{t-1})q(x_{t-1}|x_0)}{q(x_t|x_0)}.$$

$$q(x_{t-1}|x_t, x_0) = \mathcal{N}(x_{t-1}; \tilde{\mu}_t, \tilde{\beta} I).$$

$$\tilde{\mu}_t = \frac{\sqrt{\alpha_t}\bar{\beta}_{t-1}}{\bar{\beta}_t} x_t + \frac{\sqrt{\alpha_{t-1}}\beta_t}{\bar{\beta}_t} x_0 = \frac{1}{\sqrt{\alpha_t}}(x_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon).$$

NYU

Reverse Process

$$q(x_{t-1}|x_t, x_0) = \frac{q(x_t|x_{t-1})q(x_{t-1}|x_0)}{q(x_t|x_0)}.$$

$$q(x_{t-1}|x_t, x_0) = \mathcal{N}(x_{t-1}; \tilde{\mu}_t, \tilde{\beta} I).$$

$$\tilde{\mu}_t = \frac{\sqrt{\alpha_t}\bar{\beta}_{t-1}}{\bar{\beta}_t} x_t + \frac{\sqrt{\alpha_{t-1}}\beta_t}{\bar{\beta}_t} x_0 = \frac{1}{\sqrt{\alpha_t}}\left(x_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon\right).$$

- Want: train up a μ_θ to match with $\tilde{\mu}_t$.

Training

- Sometimes it is more common to predict the denoising vector ϵ instead of μ.

$$\tilde{\mu}_t = \frac{1}{\sqrt{\alpha_t}}(x_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon),$$

$$\mu_\theta(x_t, t) = \frac{1}{\sqrt{\alpha_t}}(x_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon_\theta(x_t)).$$

Training

- Sometimes it is more common to predict the denoising vector ϵ instead of μ.
- Randomly pick at a time step and predict the difference between the noisy and the original.

$$\tilde{\mu}_t = \frac{1}{\sqrt{\alpha_t}}(x_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon),$$

$$\mu_\theta(x_t, t) = \frac{1}{\sqrt{\alpha_t}}(x_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon_\theta(x_t)).$$

Training

- Sometimes it is more common to predict the denoising vector ϵ instead of μ.
- Randomly pick at a time step and predict the difference between the noisy and the original.

Algorithm 1 Training

1: **repeat**
2: $\mathbf{x}_0 \sim q(\mathbf{x}_0)$
3: $t \sim \mathrm{Uniform}(\{1, \ldots, T\})$
4: $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
5: Take gradient descent step on
$$\nabla_\theta \left\| \boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, t) \right\|^2$$
6: **until** converged

$$\tilde{\mu}_t = \frac{1}{\sqrt{\alpha_t}}(x_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon),$$

$$\mu_\theta(x_t, t) = \frac{1}{\sqrt{\alpha_t}}(x_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon_\theta(x_t)).$$

Sampling

- How do we sample an image?

Algorithm 2 Sampling

1: $\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
2: **for** $t = T, \ldots, 1$ **do**
3: $\quad \mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ if $t > 1$, else $\mathbf{z} = \mathbf{0}$
4: $\quad \mathbf{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}} \left(\mathbf{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}} \boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t) \right) + \sigma_t \mathbf{z}$
5: **end for**
6: **return** $\mathbf{x}_0$

Sampling

- How do we sample an image?
- We know μ_θ which will help us transition from x_t to x_{t-1}.

$$\mu_\theta(x_t, t) = \frac{1}{\sqrt{\alpha_t}}\left(x_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon_\theta(x_t)\right).$$

Algorithm 2 Sampling

1: $\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
2: **for** $t = T, \ldots, 1$ **do**
3: $\quad \mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ if $t > 1$, else $\mathbf{z} = \mathbf{0}$
4: $\quad \mathbf{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}}\left(\mathbf{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t)\right) + \sigma_t \mathbf{z}$
5: **end for**
6: **return** $\mathbf{x}_0$

Sampling

- How do we sample an image?
- We know μ_θ which will help us transition from x_t to x_{t-1}.

$$\mu_\theta(x_t, t) = \frac{1}{\sqrt{\alpha_t}}\left(x_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon_\theta(x_t)\right).$$

- Sample from $\mathcal{N}(x_{t-1}; \mu_\theta(x_t, t), \sigma_t^2)$. σ_t can either be β_t or $\tilde{\beta}_t$ derived from the posterior.

Algorithm 2 Sampling

$\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
for $t = T, \ldots, 1$ **do**
  $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ if $t > 1$, else $\mathbf{z} = \mathbf{0}$
  $\mathbf{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}}\left(\mathbf{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t)\right) + \sigma_t \mathbf{z}$
end for
return $\mathbf{x}_0$

More on Samplers

- DDPM relies on many iterations (e.g. 1000) to produce one sample. Slower than NFs and GANs.

[Song et al. 2021]

Song et al. Denoising Diffusion Implicit Models. ICLR 2021.

NYU

More on Samplers

- DDPM relies on many iterations (e.g. 1000) to produce one sample. Slower than NFs and GANs.
- In the non-Markovian model, we can first generate x_T, and based on x_T and x_0 we can generate x_{T-1} and so on.

[Song et al. 2021]

Song et al. Denoising Diffusion Implicit Models. ICLR 2021.

NYU

More on Samplers

- DDPM relies on many iterations (e.g. 1000) to produce one sample. Slower than NFs and GANs.
- In the non-Markovian model, we can first generate x_T, and based on x_T and x_0 we can generate x_{T-1} and so on.
- Joint distribution:

$$q(x_{1:T}|x_0) = q(x_T|x_0) \prod_{t=2}^{T} q(x_{t-1}|x_t, x_0).$$

[Song et al. 2021]

Song et al. Denoising Diffusion Implicit Models. ICLR 2021.

More on Samplers

- DDPM relies on many iterations (e.g. 1000) to produce one sample. Slower than NFs and GANs.
- In the non-Markovian model, we can first generate x_T, and based on x_T and x_0 we can generate x_{T-1} and so on.
- Joint distribution:

$$q(x_{1:T}|x_0) = q(x_T|x_0) \prod_{t=2}^{T} q(x_{t-1}|x_t, x_0).$$

- Estimate x_{t-1} based on x_0 and x_t:

$$q(x_{t-1}|x_t, x_0) = \mathcal{N}\left(\sqrt{a_{t-1}}x_0 + \sqrt{1 - \alpha_{t-1} - \sigma_t^2} \cdot \frac{x_t - \sqrt{\alpha_t}x_0}{\sqrt{1-\alpha_t}}, \sigma_t^2 I\right).$$

[Song et al. 2021]

Song et al. Denoising Diffusion Implicit Models. ICLR 2021.

More on DDIM Samplers

- Prediction of x_0:

$$f_\theta^{(t)}(x_t) = \frac{1}{\sqrt{\alpha_t}}(x_t - \sqrt{1-\alpha_t} \cdot \epsilon_\theta^{(t)}(x_t)).$$

[Song et al. 2021]

More on DDIM Samplers

- Prediction of x_0:

$$f_\theta^{(t)}(x_t) = \frac{1}{\sqrt{\alpha_t}}(x_t - \sqrt{1-\alpha_t} \cdot \epsilon_\theta^{(t)}(x_t)).$$

- Sampling process:

$$p_\theta^{(t)}(x_{t-1}|x_t) = \begin{cases} \mathcal{N}(f_\theta^{(1)}(x_t)), \sigma_1^2 I) & \text{if } t = 1 \\ q(x_{t-1}|x_t, f_\theta^{(t)}(x_t)) & \text{otherwise.} \end{cases}$$

[Song et al. 2021]

Guided Diffusion

- We can add guidance on the diffusion updates at inference time.

Classifier Guidance / External Score Model

$$x_{t-1} \leftarrow \text{sample from } \mathcal{N}(\mu + s\Sigma \nabla_{x_t} \log p_\phi(y|x_t), \Sigma)$$

$$\hat{\epsilon} \leftarrow \epsilon_\theta(x_t) - \sqrt{1-\bar{\alpha}_t}\, \nabla_{x_t} \log p_\phi(y|x_t)$$

$$x_{t-1} \leftarrow \sqrt{\bar{\alpha}_{t-1}} \left(\frac{x_t - \sqrt{1-\bar{\alpha}_t}\hat{\epsilon}}{\sqrt{\bar{\alpha}_t}} \right) + \sqrt{1-\bar{\alpha}_{t-1}}\hat{\epsilon}$$

Dhariwal & Nichol. Diffusion Models Beat GANs on Image Synthesis. NeurIPS 2021.
Ho & Salimans. Classifier-Free Diffusion Guidance. 2022.

NYU

Guided Diffusion

- We can add guidance on the diffusion updates at inference time.

Classifier Guidance / External Score Model

$$x_{t-1} \leftarrow \text{sample from } \mathcal{N}(\mu + s\Sigma \nabla_{x_t} \log p_\phi(y|x_t), \Sigma)$$

$$\hat{\epsilon} \leftarrow \epsilon_\theta(x_t) - \sqrt{1-\bar{\alpha}_t}\, \nabla_{x_t} \log p_\phi(y|x_t)$$

$$x_{t-1} \leftarrow \sqrt{\bar{\alpha}_{t-1}} \left(\frac{x_t - \sqrt{1-\bar{\alpha}_t}\hat{\epsilon}}{\sqrt{\bar{\alpha}_t}} \right) + \sqrt{1-\bar{\alpha}_{t-1}}\hat{\epsilon}$$

- We also can train a conditional diffusion model.

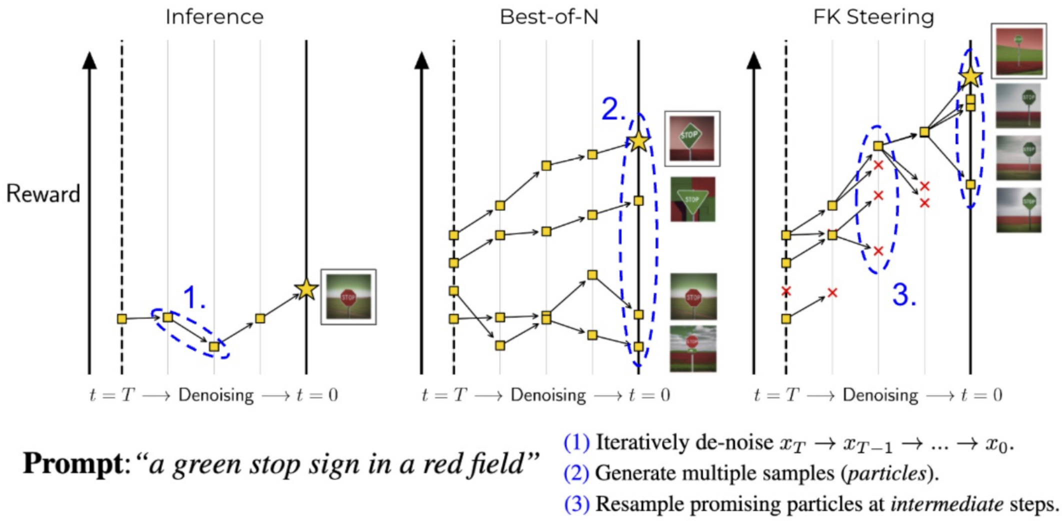
repeat
  $(\mathbf{x}, \mathbf{c}) \sim p(\mathbf{x}, \mathbf{c})$ ▷ Sample data with conditioning from the dataset
  $\mathbf{c} \leftarrow \varnothing$ with probability p_{uncond} ▷ Randomly discard conditioning to train unconditionally
  $\lambda \sim p(\lambda)$ ▷ Sample log SNR value
  $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
  $\mathbf{z}_\lambda = \alpha_\lambda \mathbf{x} + \sigma_\lambda \boldsymbol{\epsilon}$ ▷ Corrupt data to the sampled log SNR value
  Take gradient step on $\nabla_\theta \left\| \boldsymbol{\epsilon}_\theta(\mathbf{z}_\lambda, \mathbf{c}) - \boldsymbol{\epsilon} \right\|^2$ ▷ Optimization of denoising model
until converged

Dhariwal & Nichol. Diffusion Models Beat GANs on Image Synthesis. NeurIPS 2021.
Ho & Salimans. Classifier-Free Diffusion Guidance. 2022.

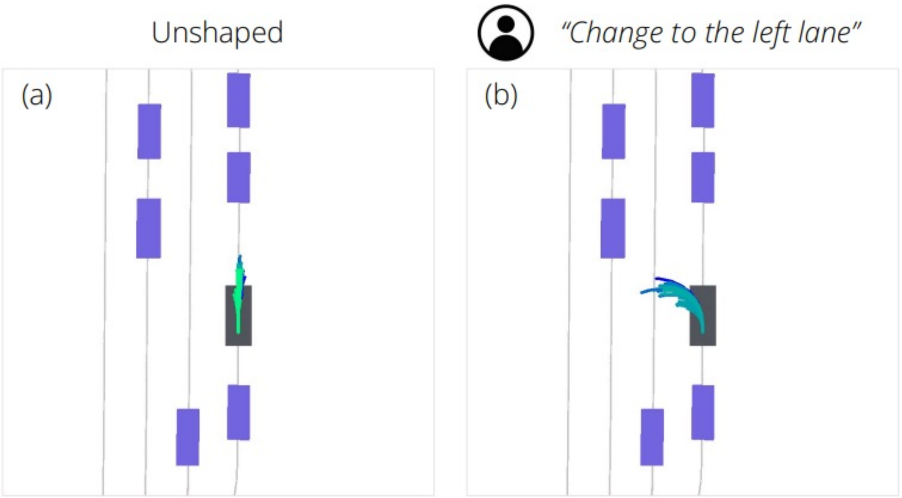
NYU

Test-Time Adaptation

- Diffusion can be combined / guided with reward functions at test time.



[Singhal et al. 2025]



[Yang et al. 2024]

Singhal et al. A General Framework for Inference-time Scaling and Steering of Diffusion Models. arXiv 2025.
Yang et al. Diffusion-ES: Gradient-free Planning with Diffusion for Autonomous and Instruction-guided Driving. CVPR 2024.

Diffusion for Detection

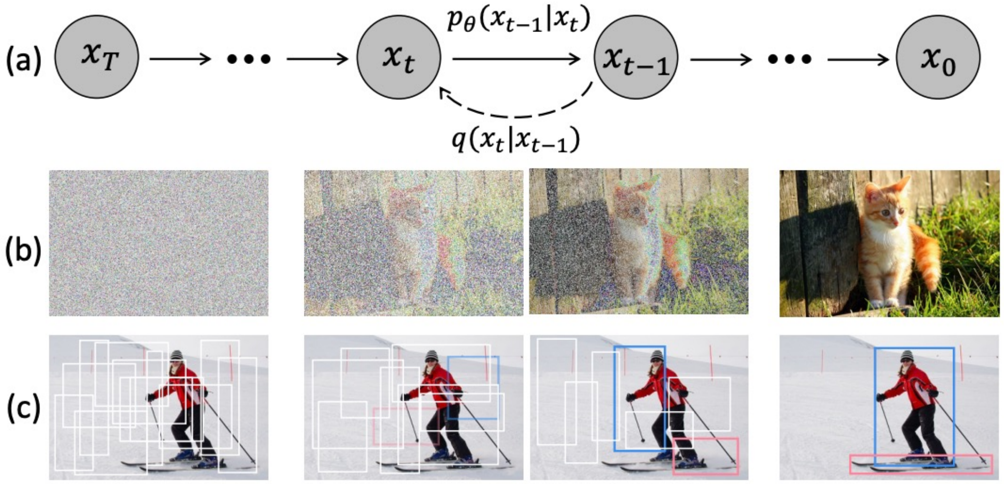
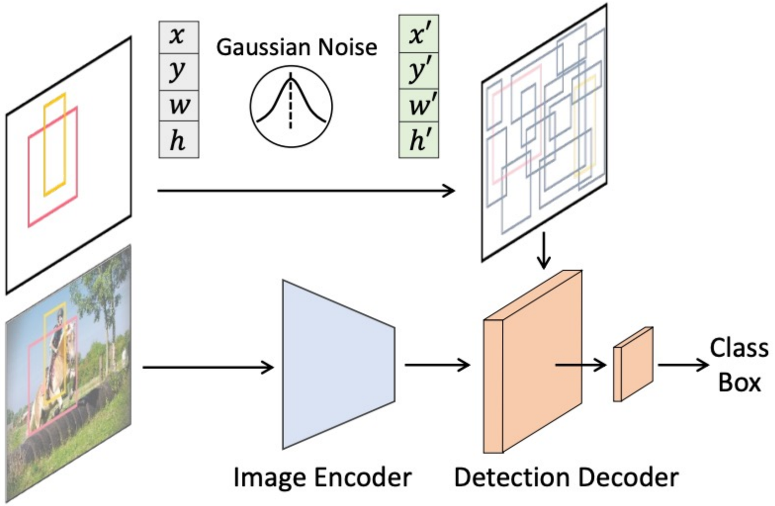
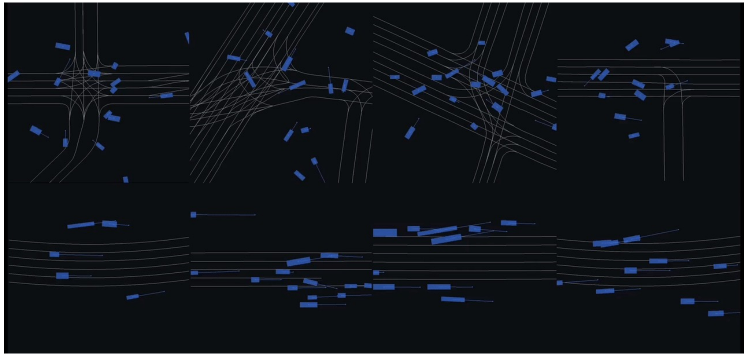


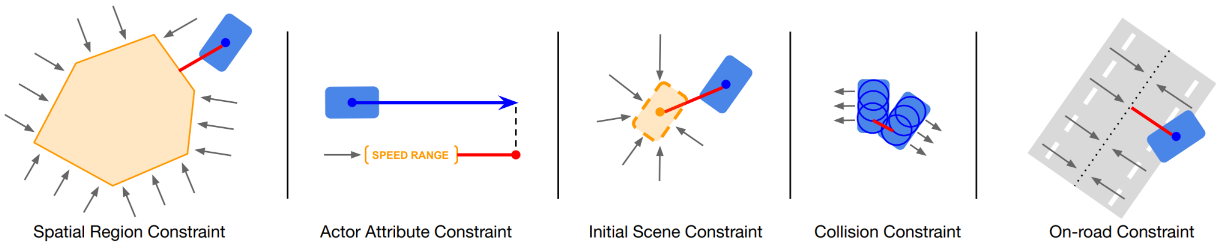
Figure 1. **Diffusion model for object detection**. (a) A diffusion model where q is the diffusion process and p_θ is the reverse process. (b) Diffusion model for image generation task. (c) We propose to formulate object detection as a denoising diffusion process from noisy boxes to object boxes.

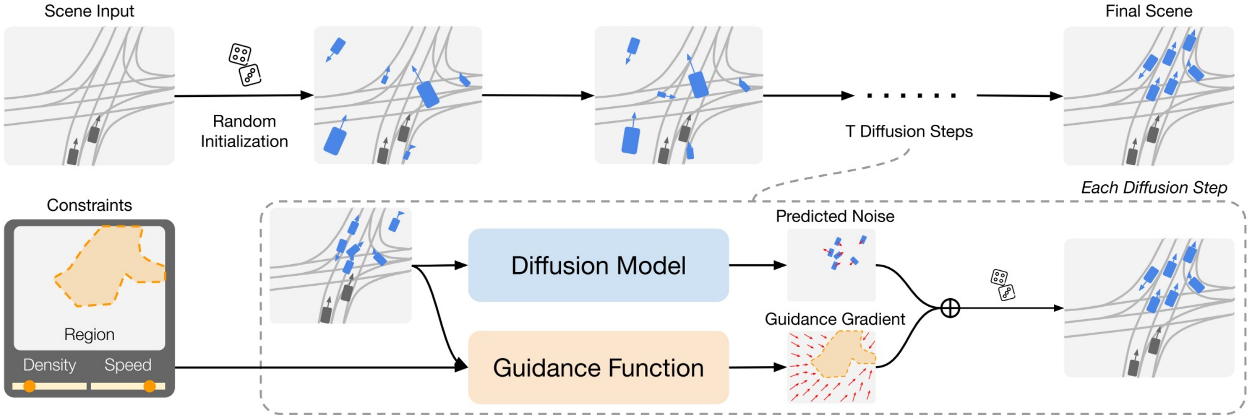


Chen et al. DiffusionDet: Diffusion Model for Object Detection. ICCV 2023.

Diffusion for Generating Simulation Scenes

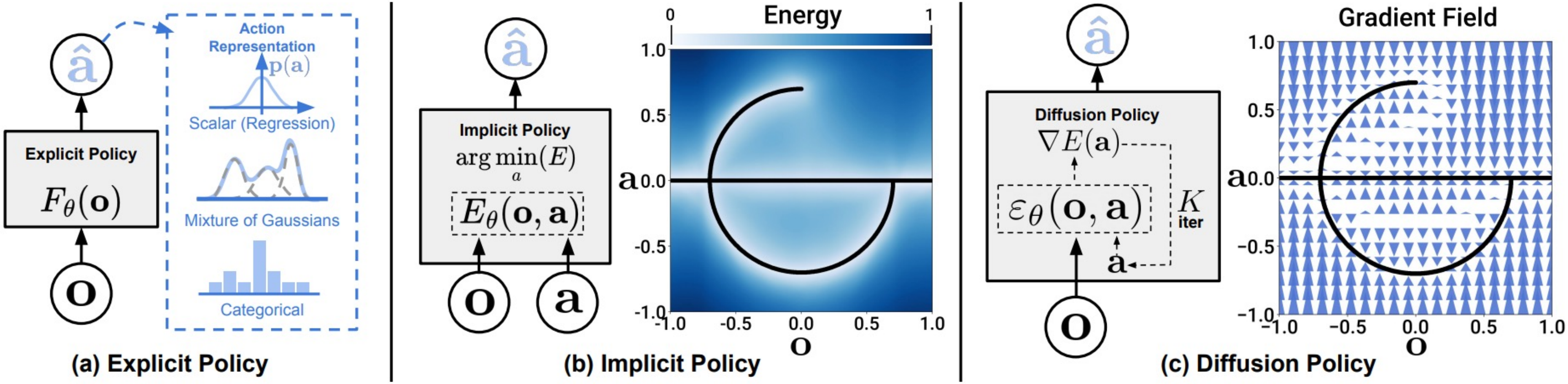






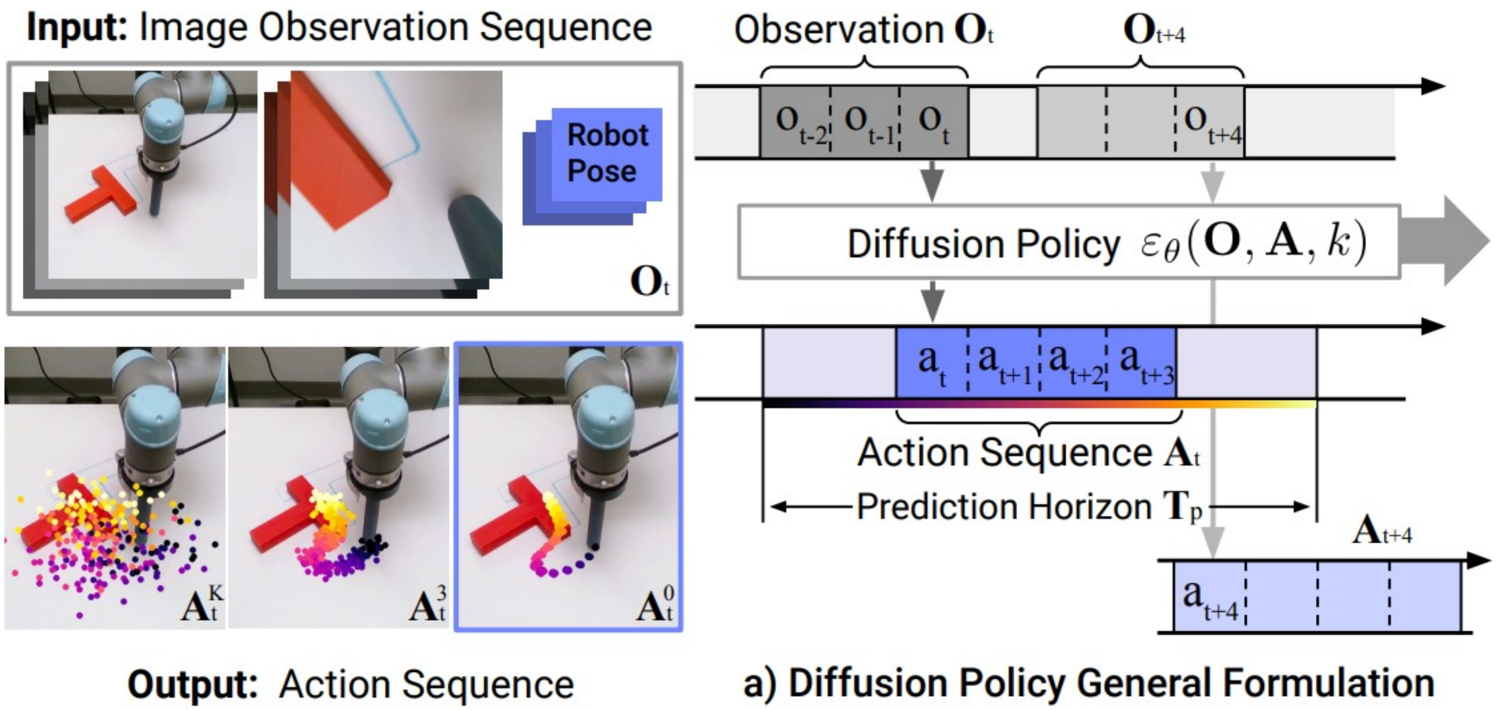
Lu et al. SceneControl: Diffusion for Controllable Traffic Scene Generation. ICRA 2024.

Diffusion for Planning and Control



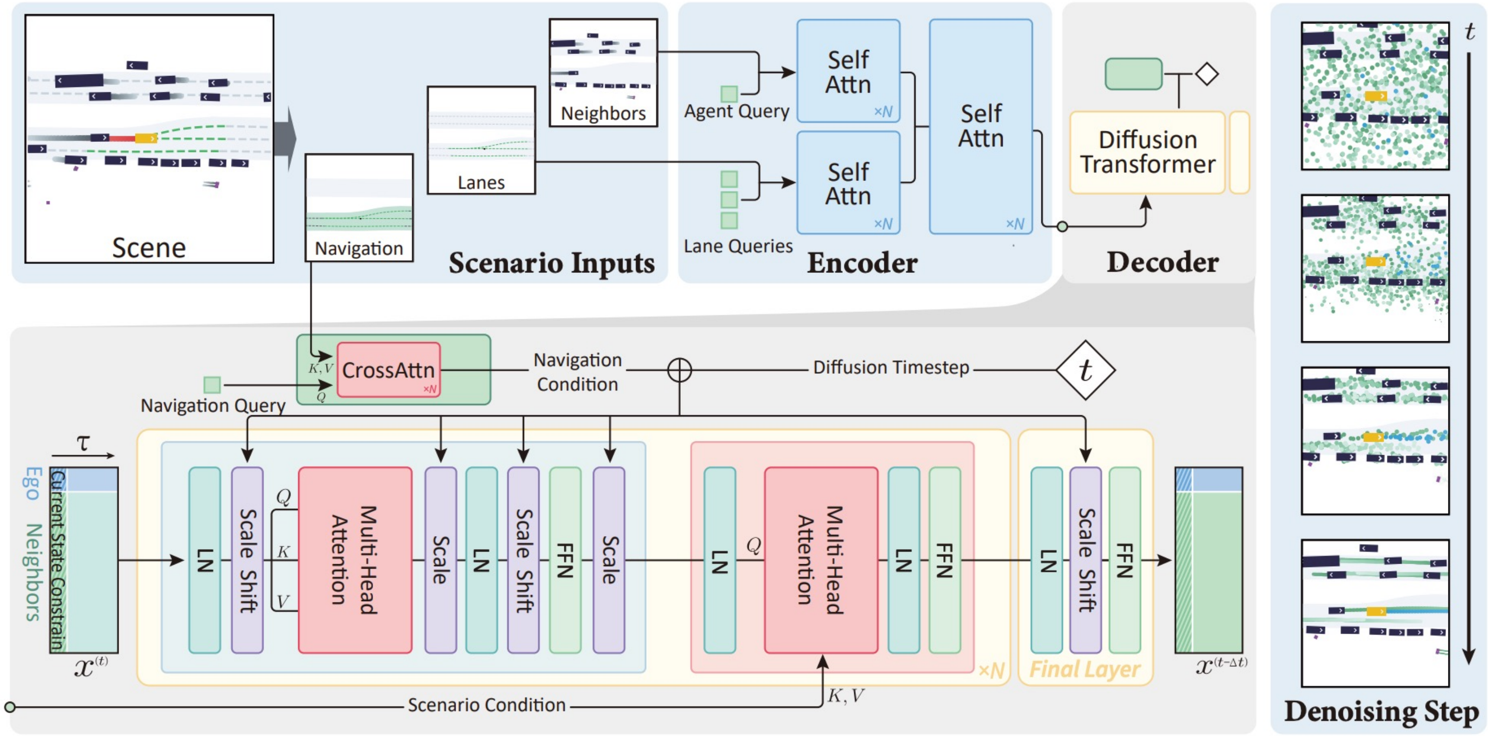
Chi et al. Diffusion Policy: Visuomotor Policy Learning via Action Diffusion. RSS 2023.

Diffusion for Planning and Control



Chi et al. Diffusion Policy: Visuomotor Policy Learning via Action Diffusion. RSS 2023.

Diffusion Planner for Self-Driving



https://openreview.net/forum?id=wM2sfVgMDH

Diffusion-Based Planning for Autonomous Driving with Flexible Guidance. OpenReview 2024.

Summary: DL for Structured Outputs

Summary: DL for Structured Outputs

- Expanding the output dimension has limitations.

Summary: DL for Structured Outputs

- Expanding the output dimension has limitations.
- Requires us thinking about generative models.
  - Graphical models
  - Autoregressive
  - Energy-based
  - Diffusion

Summary: DL for Structured Outputs

- Expanding the output dimension has limitations.
- Requires us thinking about generative models.
  - Graphical models
  - Autoregressive
  - Energy-based
  - Diffusion
- Understand pros and cons. Experiment with each option.

Summary: DL for Structured Outputs

- Expanding the output dimension has limitations.
- Requires us thinking about generative models.
  - Graphical models
  - Autoregressive
  - Energy-based
  - Diffusion
- Understand pros and cons. Experiment with each option.
- Application in embodied environments.

What's Next

- Tutorial on simulation environments
- Next week: 3D vision, mapping