Online ensemble tracking with CNNs: challenges
- Lack of training data
- Lack of diversity of CNN models
- Label noise of training examples

Main contribution
- BranchOut: regularization for ensemble CNNs
  - Well-suited for online ensemble learning
  - Stochastically selecting updating branches
  - Related to DropOut [1] and DropConnect [2]
- Sharing idea with bagging
- Multi-level target representation using a CNN
  - Implementing with multi-level FC layers
  - Representing targets with variable abstraction levels
- Well-organized experiment
  - SOTA performance in the standard datasets
  - Proper ablation study

Architecture
- A variation on the award-winning MDNet [3] architecture
- Multiple branches with a variable number of FC layers
  - 10 branches: 5 with one layer and 5 with two layers
- Ensemble and classification layers

Stochastic learning with regularization
- Stochastic branch selection:
  - \( a_k \sim \text{Bernoulli}(p_k) \)
- Loss function from all branches:
  \[
  L = \sum_{i=1}^{K} \sum_{k=1}^{K} a_k \mathcal{L}(F_i(x'; \theta_k))
  \]
- Stochastic gradient: updating FC layers only
  \[
  \frac{\partial L}{\partial \theta_k} = \sum_{i=1}^{K} \sum_{k=1}^{K} a_k \frac{\partial}{\partial \theta_k} \mathcal{L}(F_i(x'; \theta_k))
  \]

Tracking algorithm and implementation details
- CNN initialization
  - Conv1-3 layers: from VGG-M
  - FC layers: random (learned with GT at the 1st frame)
- Main loop
  - Dense sampling (256) for observation
  - Track in translation+scale space
  - Target estimation:
    - Additional components
      - Bounding box regression
      - Hard negative mining
      - Search space expansion upon failure
      - Model update
        - Long- and short-term updates [3]
        - Selected branches only

Algorithm I: Stochastic ensemble tracking by BranchOut
Requirer: CNN with K branches of FC layers parameterized by \( \theta = (\theta_1, \ldots, \theta_K) \) and initial target state \( x_1 \).

- Target states \( x_t \):
  1. Randomly initialize \( \theta = (\theta_1, \ldots, \theta_K) \).
  2. Train a bounding box regression model.
  3. Draw positive samples \( S_t^+ \) and negative samples \( S_t^- \).
  4. Update \( \theta_t \) using \( S_t^+ \) and \( S_t^- \).
  5. \( t = t + 1 \).
  6. end if

- Draw target candidate samples \( x_t \in [1, \ldots, N] \).
  7. Find the optimal target state \( \hat{x}_t \) by Eq. (16).
    - if \( F_t(\hat{x}_t) > 0.6 \) then
    8. Draw new training samples \( S_t^+ \) and \( S_t^- \).
    9. \( T = T + 1 \).
  10. if \( T > T' \) then
    11. end if
    12. \( T' = T' \times (\text{init}, \text{ens}) \).

- Model update
  - Long- and short-term updates [3]
  - Selected branches only

BranchOut: Diversifying the models learned in individual branches
Avoiding contamination of models by partial updates
Less computation compared to naive ensemble

BranchOut (with Model I)

- Learning Rate: \( 0.0005 \times 0.98 \) every 500 frames
- update: \( \text{Cyclical} \), \( \text{Cyclical} \), \( \text{Cyclical} \), \( \text{Cyclical} \)
- Main loop: \( \text{template} \), \( \text{template} \), \( \text{template} \), \( \text{template} \)
- Accuracy: \( 0.683 \), \( 0.591 \), \( 0.679 \), \( 0.674 \)
- Precision (\#top 50): \( 0.640 \), \( 0.643 \), \( 0.669 \), \( 0.642 \)

VOT2015 results

- Contribution of stochastic update and multi-level representation

BranchOut (with Model I)

- Learning Rate: \( 0.0005 \times 0.98 \) every 500 frames
- update: \( \text{Cyclical} \), \( \text{Cyclical} \), \( \text{Cyclical} \), \( \text{Cyclical} \)
- Main loop: \( \text{template} \), \( \text{template} \), \( \text{template} \), \( \text{template} \)
- Accuracy: \( 0.683 \), \( 0.591 \), \( 0.679 \), \( 0.674 \)
- Precision (\#top 50): \( 0.640 \), \( 0.643 \), \( 0.669 \), \( 0.642 \)

References