Probabilistic Attention for Interactive Segmentation

Prasad Gabbur, Manjot Bilkhu, Javier Movellan | NeurIPS 2021

Overview

Attention as probabilistic inference in a generative model

- Standard dot-product attention as a special case

Online Inferences from the model

- Online Key Adaptation
- Online Value Propagation

Experiments

- Interactive Segmentation

Overview

Attention as probabilistic inference in a generative model

- Standard dot-product attention as a special case

Online Inferences from the model

- Online Key Adaptation
- Online Value Propagation

Experiments

- Interactive Segmentation

Overview

Attention as probabilistic inference in a generative model

- Standard dot-product attention as a special case

Online Inferences from the model

- Online Key Adaptation
- Online Value Propagation

Experiments

- Interactive Segmentation

Probabilistic Model

Joint distribution of query (q), value (v) pair at token i as a mixture over n latent memory units (u_j)

$$p_i(q, v) = \sum_{j=1}^{n} p_i(q, v, u_j) = \sum_{j=1}^{n} \pi_{i,j} p_i(q, v \mid u_j)$$

Each token's (query, value) pair is generated i.i.d. from the mixture

$$p(q_{1:n}, v_{1:n}) = \prod_{i=1}^{n} p_i(q_i, v_i)$$

Attention: Maximum A Posteriori (MAP) Value Inference

$$\hat{v} = \arg\max_{v} p_i(v \mid q)$$

Special Case: Dot-Product Attention

Assume conditional independence and isotropic Gaussian likelihoods for per-unit query and value marginals

$$p_i(q, v) = \sum_{j=1}^{n} \pi_{i,j} p_i(q, v \mid u_j)$$

Parameters

ξ_j: Query Gaussian mean (Key at unit (token) j)

α_j: Query Gaussian precision

μ_j: Value Gaussian mean (Value at unit (token) j)

β_j: Value Gaussian precision

$$p_i(q, v \mid u_j) = p_i(q \mid u_j) p_i(v \mid u_j)$$

$$p_i(q \mid u_j) = \left(\frac{\alpha_j}{2\pi}\right)^{\frac{d}{2}} e^{-\frac{\alpha_j}{2}\|q-\xi_j\|^2}$$

$$p_i(v \mid u_j) = \left(\frac{\beta_j}{2\pi}\right)^{\frac{m}{2}} e^{-\frac{\beta_j}{2}\|v-\mu_j\|^2}$$

Special Case: Dot-Product Attention

Let

$$\pi_{i,j} \propto \exp(\|\xi_j\|^2)\exp(\|\mu_j\|^2)$$

$$\alpha_1 = \alpha_2 = \ldots = \alpha_n = \alpha$$

$$\beta_1 = \beta_2 = \ldots = \beta_n = \beta \to 0$$

MAP Value Inference with Expectation Maximization (EM)

$$\hat{v} = \arg\max_v p_i(v \mid q)$$

$$= \sum_j \left(\frac{e^{\alpha \xi_j^T q}}{\sum_k e^{\alpha \xi_k^T q}} \right) \mu_j$$

(Proof: Appendix B.1 in the paper)

Online Inferences

Online Key Adaptation based on observed queries

$$\hat{v} = \arg\max_{v} p_i(v \mid q_{1:n}) \longrightarrow p_i(v \mid q_{1:n}) = \int p(\xi_{1:n} \mid q_{1:n}) p_i(v \mid q_i, \xi_{1:n}) d\xi_{1:n} \approx p_i(v \mid q_i, \hat{\xi}_{1:n})$$

$$\hat{\xi}_{1:n} = \arg\max_{\xi_{1:n}} p(\xi_{1:n} \mid q_{1:n})$$

(Query likelihood parameters)

Online Value Propagation based on user feedback (Interactive Segmentation)

$$\hat{v}_i = \arg\max_{v} p_i(v \mid q_{1:n}, v_{1:s}) \longrightarrow p_i(v \mid q_{1:n}, v_{1:s}) = \int p(\mu_{1:n} \mid q_{1:n}, v_{1:s}) p_i(v \mid q_i, \mu_{1:n}) d\mu_{1:n} \approx p_i(v \mid q_i, \hat{\mu}_{1:n})$$

$$s < i \leq n$$

$$\hat{\mu}_{1:n} = \arg\max_{\mu_{1:n}} p(\mu_{1:n} \mid q_{1:n}, v_{1:s})$$

(Value likelihood parameters)

Online Key Adaptation

Adapt mixture model parameters ($\xi_{1:n}, \alpha_{1:n}$) based on observed queries before doing value inference

Key Adaptation

$$\hat{\xi}_{1:n} = \arg\max_{\xi_{1:n}} p(\xi_{1:n} \mid q_{1:n})$$

$$\xi_k^{t+1} = \frac{\theta_\xi \xi_k^t + \alpha_k \sum_{i=1}^n w_{i,k}^t q_i}{\theta_\xi + \alpha_k \sum_{i=1}^n w_{i,k}^t}$$

EM update with a Gaussian prior $\mathcal{N}(\xi_k \mid \xi_k^0, \theta_\xi^{-1} I)$ to control overfitting

$$w_{i,j}^t = \frac{\pi_{i,j}\, p(q_i \mid u_j, \xi_j^t)}{\sum_{k=1}^n \pi_{i,k}\, p(q_i \mid u_k, \xi_k^t)}$$

(Equation 18 & Appendix B.2 in the paper)

Online Value Propagation

Propagate user feedback (fixed values) for some units to others through updates to value model parameters $(\mu_{1:n}, \beta_{1:n})$

Value Propagation

$$\hat{\mu}_{1:n} = \arg\max_{\mu_{1:n}} p(\mu_{1:n} \mid q_{1:n}, v_{1:s})$$

$$\mu_k^{t+1} = \frac{\theta_\mu \mu_k^t + \beta_k \sum_{i=1}^{s} w_{i,k}^t v_i}{\theta_\mu + \beta_k \sum_{i=1}^{s} w_{i,k}^t}$$

EM update with a Gaussian prior $\mathcal{N}(\mu_k \mid \mu_k^0, \theta_\mu^{-1} I)$ to control overfitting and/or scale feedback

$$w_{i,k}^t = \frac{\pi_{i,k}\ p(q_i \mid u_k, \xi_k) p(v_i \mid u_k, \mu_k^t)}{\sum_{j=1}^{n} \pi_{i,k} p(q_i \mid u_j, \xi_j) p(v_i \mid u_j, \mu_j^t)}$$

(Equation 23 & Appendix B.4 in the paper)

Online Inferences and Supervised Training

Supervised Training

Query, Key and Value embedding parameters are usually learnt through gradient descent on task-specific loss function(s)

Online Inferences

Online Key adaptation and Value propagation's

parameters $(\alpha_j, \beta_j, \pi_{i,j}, \theta_\xi, \theta_\mu)$ can be chosen through cross-validation

Proposed model allows back-propagating through online inferences during supervised training

Position Embeddings

Allow attention models to encode the relative or absolute positions of tokens, critical for computer vision

Introduce extra parameters within the per-unit mixture likelihoods and priors

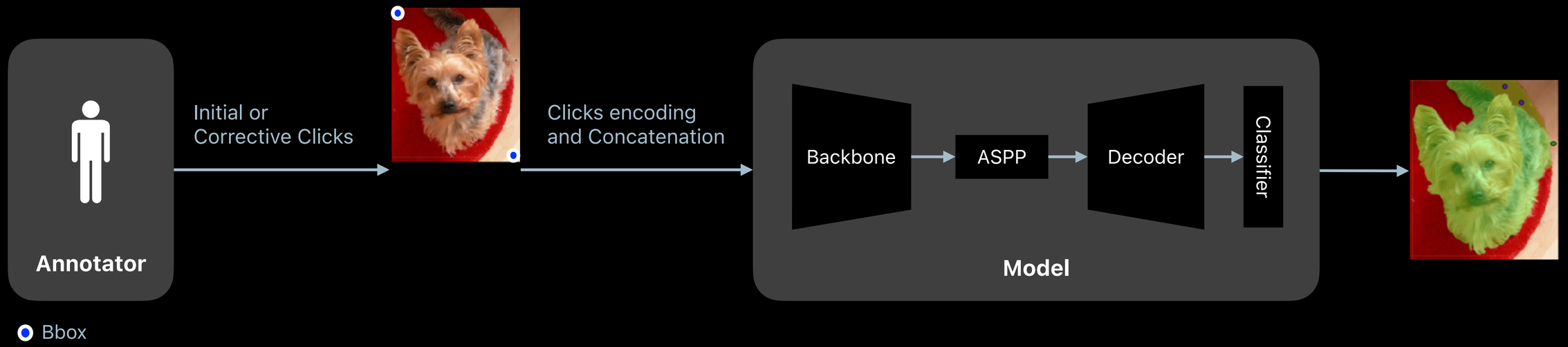
$$p_i(q \mid \xi_j, r^q_{j-i}, u_j) \propto \mathcal{N}(q \mid \frac{\xi_j + r^q_{j-i}}{2}, \frac{1}{2\alpha_j} I)$$

r^q_{j-i} and r^k_{j-i} encode relative position embeddings of queries and keys respectively

$$\pi_{i,j} \propto \mathcal{N}(\xi_j \mid r^k_{j-i}, \frac{1}{\alpha_j} I) \exp\left[\frac{\alpha_j}{2}\left(2\|\xi_j\|^2 + \|r^q_{j-i}\|^2 + \|r^k_{j-i}\|^2\right)\right] \exp\left[\frac{\beta_j}{2}\|\mu_j\|^2\right]$$

Experiments

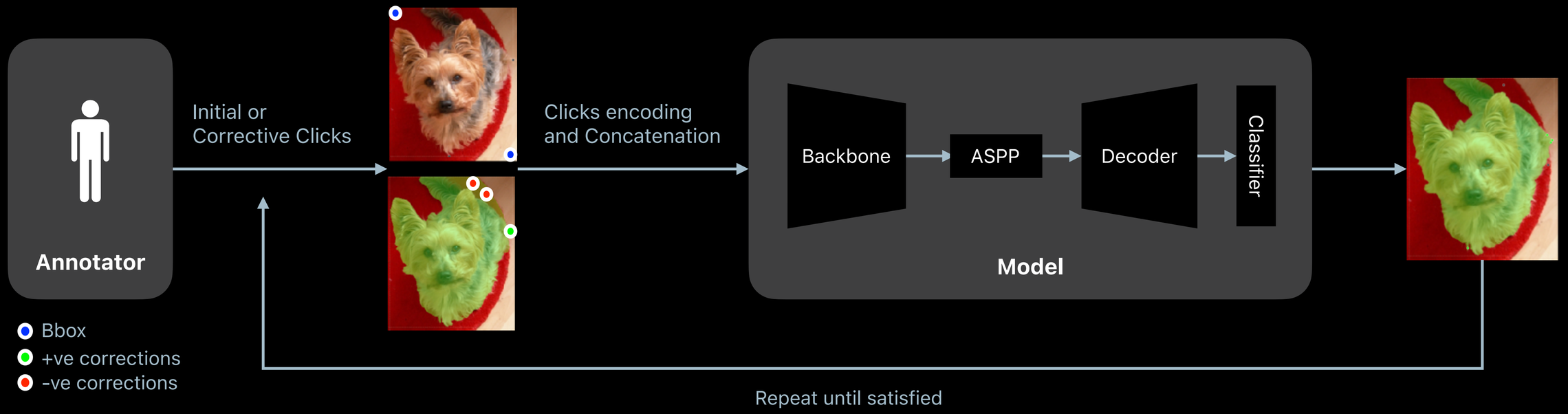
Interactive Segmentation



- DeepLabV3 style architecture for interactive segmentation
- Curriculum learning over 3 tasks (Appendix C.2 in the paper)
- Train on LVIS or SBD and test on GrabCut, Berkeley (standard benchmarks)

Experiments

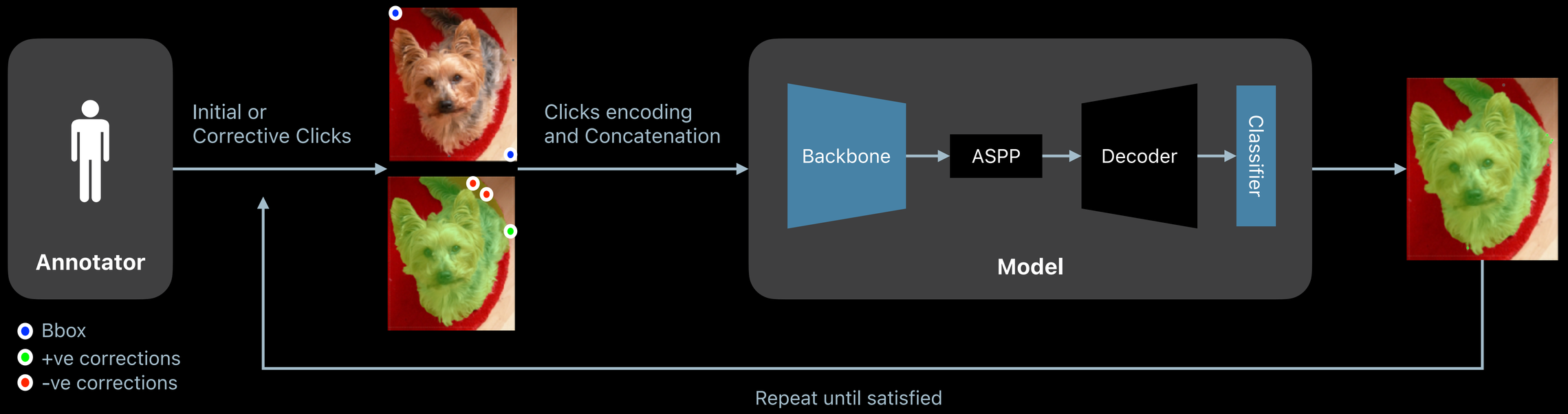
Interactive Segmentation



- DeepLabV3 style architecture for interactive segmentation
- Curriculum learning over 3 tasks (Appendix C.2 in the paper)
- Train on LVIS or SBD and test on GrabCut, Berkeley (standard benchmarks)

Experiments

Interactive Segmentation



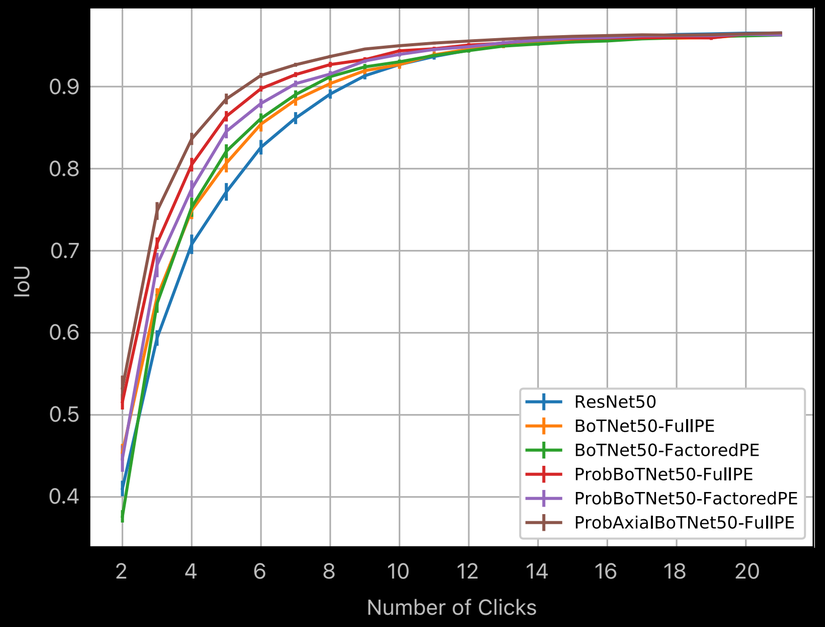
- Probabilistic Attention within the backbone
- Probabilistic Attention within the classifier
- Both of the above

Probabilistic Attention within the Backbone

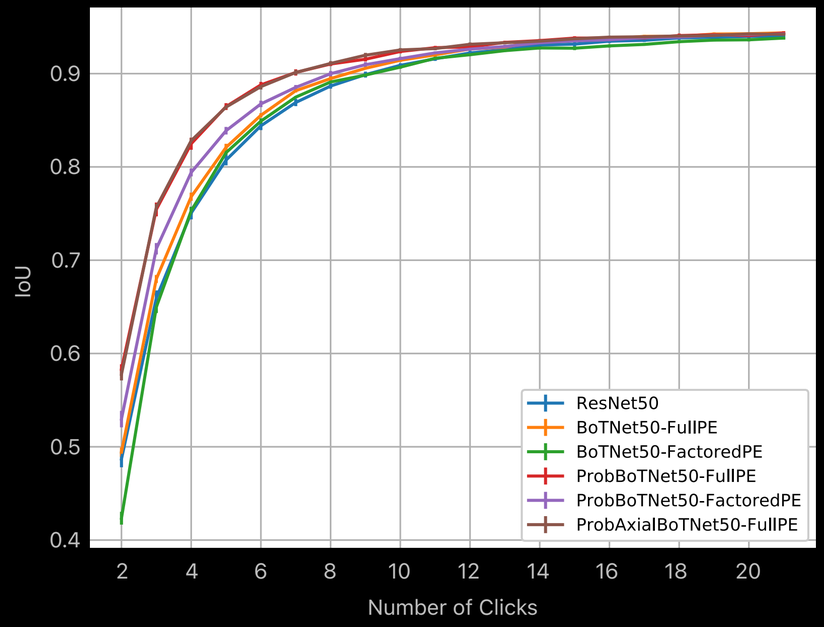
Replace standard dot-product attention with probabilistic attention within the BoTNet50* architecture used as the backbone

- Factored and Full Position Embeddings (PE), Full and Axial Attention
- Plot IoU gain per click (higher => more responsive models)

GrabCut



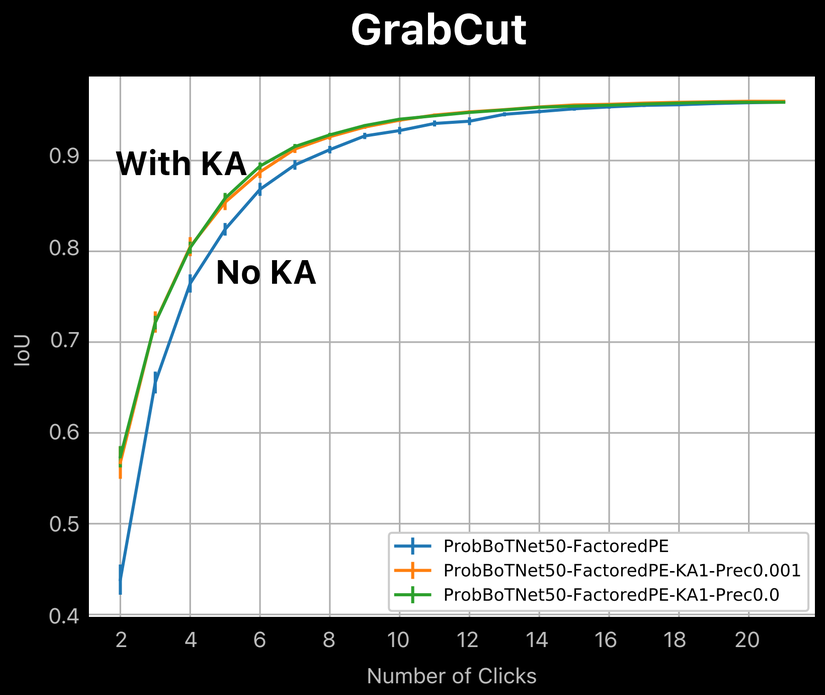
Berkeley

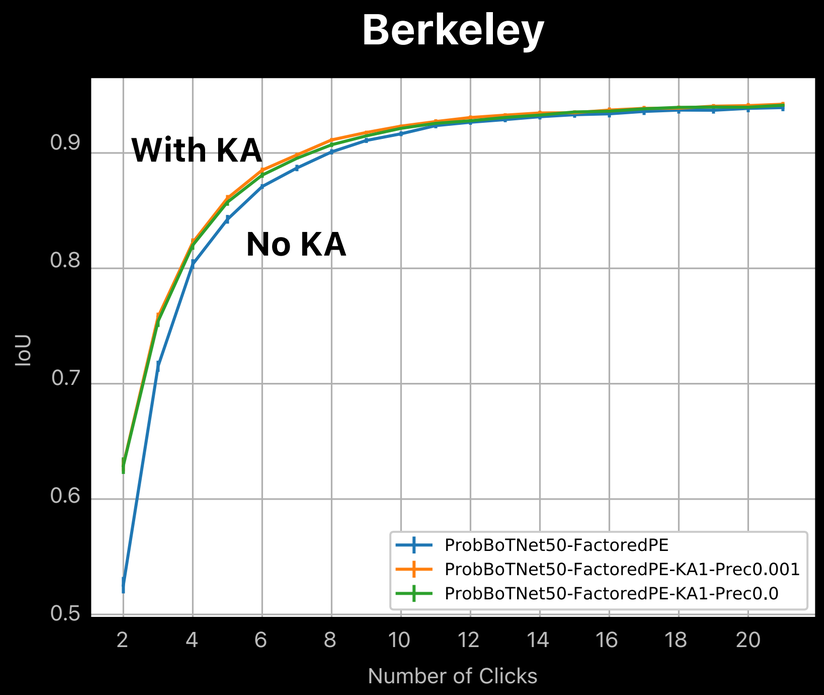


* Srinivas et. al., Bottleneck Transformers for Visual Recognition, CVPR 2021

Probabilistic Attention within the Backbone

Effect of Key Adaptation (KA) within the BoTNet50 backbone
Improves model performance with no or small annotator feedback

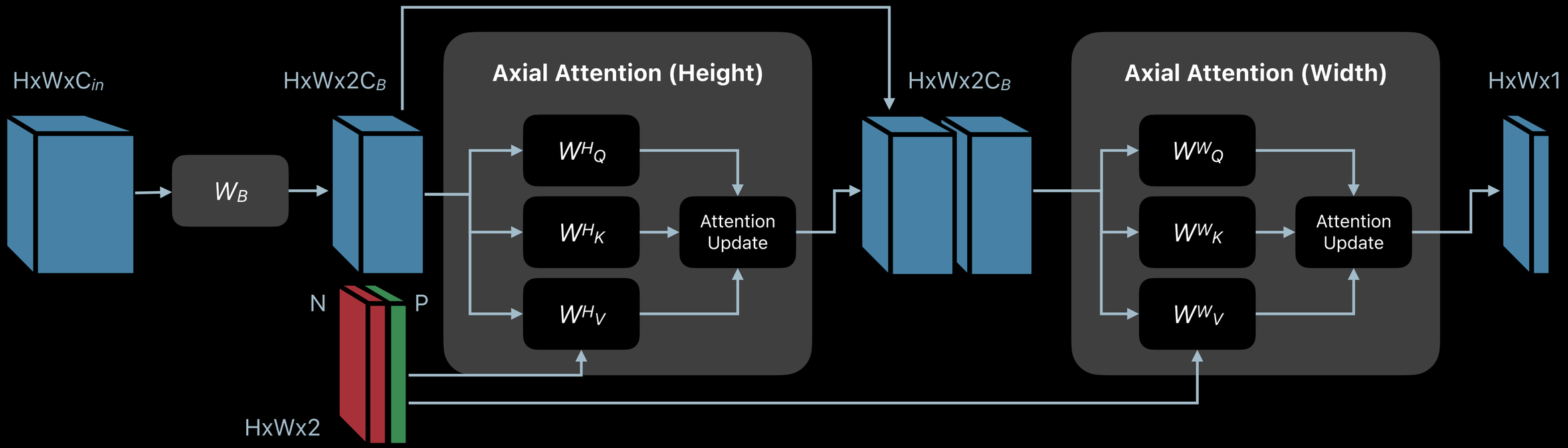




Probabilistic Attention within the Classifier

Replace 1x1 convolution classifier head with a self-attention based classifier

Corrective Self-Attention (CSA) Classifier

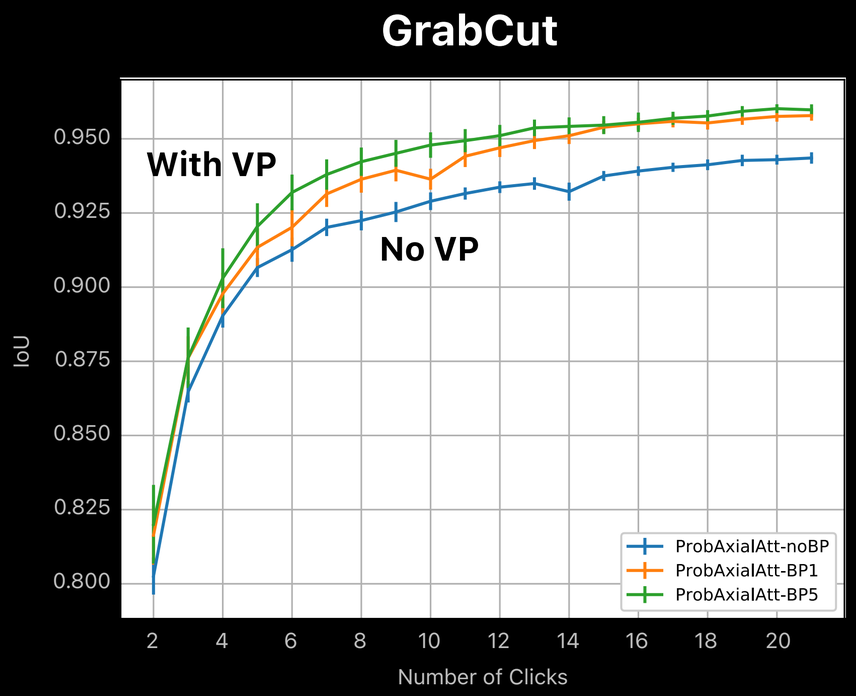


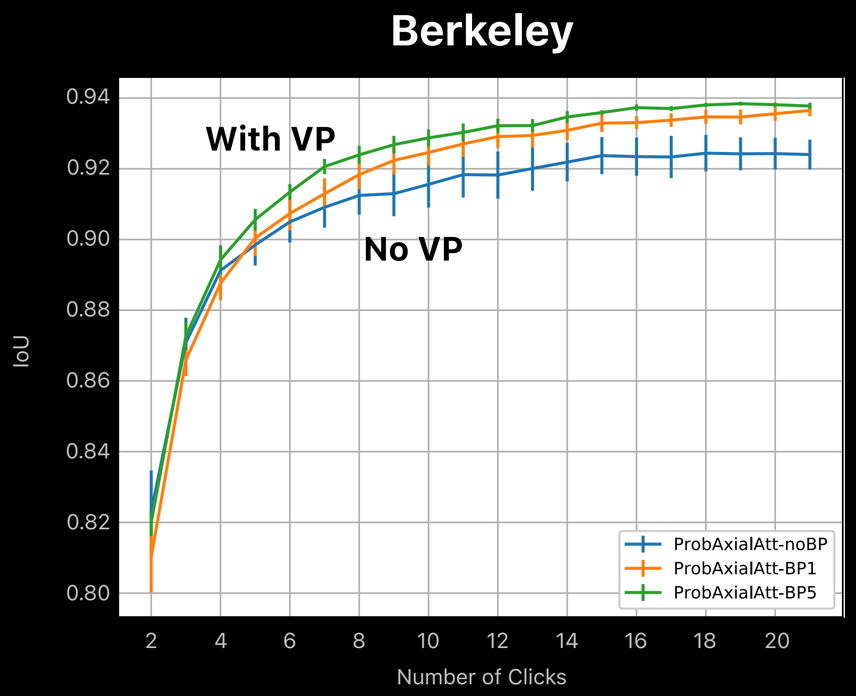
- Axial Attention enables working at image resolution
- Use probabilistic attention for the update

Probabilistic Attention within the Classifier

Effect of Value Propagation (VP) within the CSA Classifier

Improves model responsiveness, especially with more annotator feedback



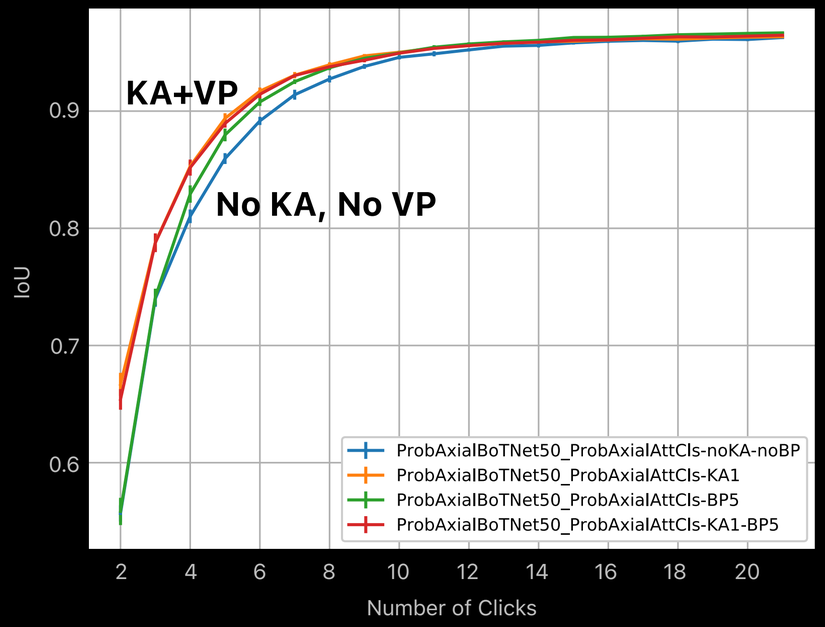


BP = VP = Value Propagation

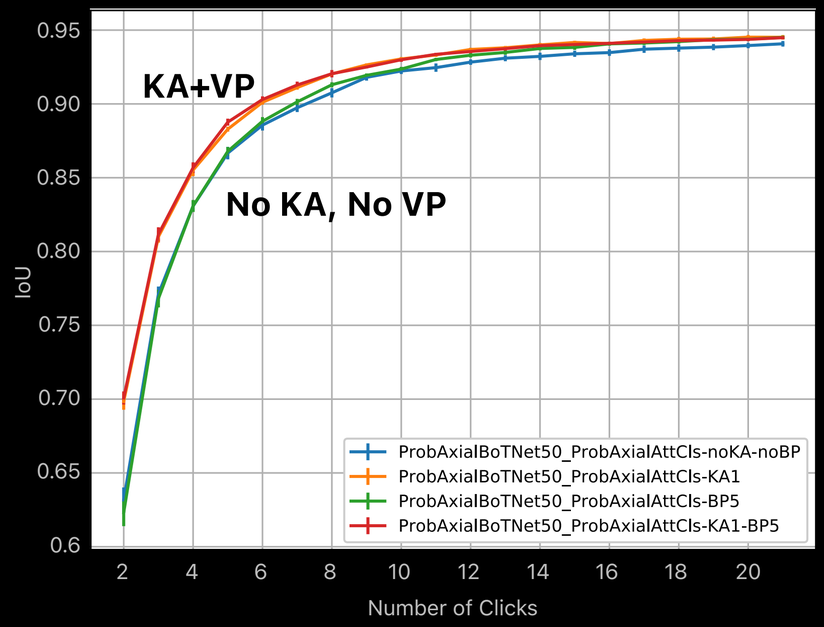
Combining Key Adaptation and Value Propagation

Effect of Key Adaptation (BoTNet50) and Value Propagation (CSA)
Improves performance in both the low and high annotator feedback regimes

GrabCut



Berkeley



KA = Key Adaptation, BP = VP = Value Propagation

Conclusions

- Probabilistic generative model for attention
- Bayesian Inferences
  - Online Key Adaptation
  - Online Value Propagation
- Applications to Interactive Segmentation
  - Extend to other domains by replacing current forms of attention
- PyTorch implementation of our model

https://github.com/apple/ml-probabilistic-attention