Stanford
ARTIFICIAL
INTELLIGENCE

Model-Based Preference Optimization:

Active Learning

Sang Truong

All Sections

Export PDF

Stanford
ARTIFICIAL
INTELLIGENCE

0. Review on Supervised Learning

- In supervised learning, we have a dataset $\mathcal{D} = \{(x_1, y_1), \ldots, (x_N, y_N)\}$ where x_i is the input and y_i is the output.
- Without the loss of generality, we concern the binary classification setting, where $y_i \in \{0, 1\}$. This fits naturally into the scope of the course on preference learning.
- The goal is to learn a model f that maps inputs to outputs, i.e., $f : x \to y$.
- In passive learning, a dataset is given, and the model is trained on the entire dataset to minimize the loss function $L(f(x_i), y_i)$.
- This is not always feasible as labeling data can be expensive or time-consuming.

Stanford
ARTIFICIAL
INTELLIGENCE

1. Introduction to Active Learning

- Active learning (AL) is a machine learning paradigm that aims to reduce the amount of labeled data required to train a model to achieve high accuracy.
- Toward this goal, AL algorithms aim to iteratively select an input datapoint for an oracle (e.g., a human annotator) to label such that when the label is observed, the model improves the most.

Stanford
ARTIFICIAL
INTELLIGENCE

1. Introduction to Active Learning

There are two primary setups in active learning:

- **Pool-based:** The model selects samples from a large unlabeled pool of data. For example, a model for text classification selects the most uncertain texts from a large pool to ask a human annotator to label.
- **Stream-based:** The model receives samples sequentially (one sample at a time) and decides whether to label them. The data is gone if the decision maker decides not to label it. For example, a system monitoring sensor data decides on-the-fly whether new sensor readings are valuable enough to label.

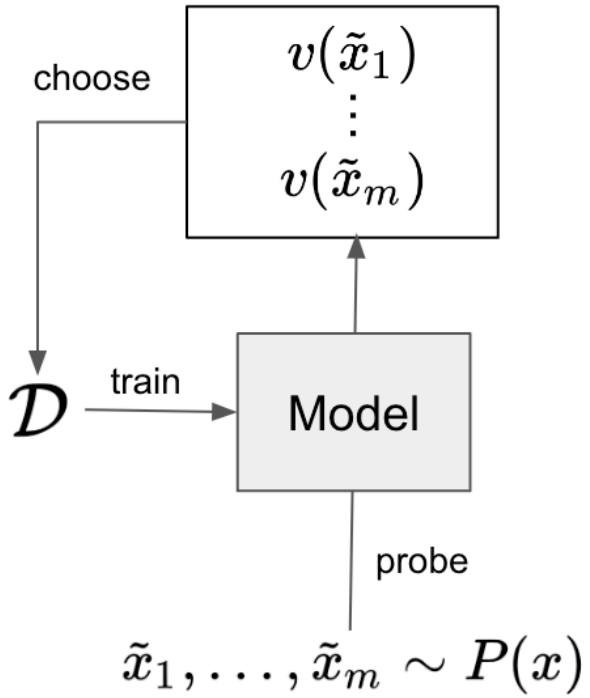
Stanford
ARTIFICIAL
INTELLIGENCE

1. Introduction to Active Learning

- Current model trained on current dataset $\mathcal{D}$, potential points $\tilde{x}_1 \ldots \tilde{x}_m$ are being investigated. AL will choose one of them to add to the dataset.
- Relative to the model, a proxy highlights the relative value of each point to model improvement $(v(\tilde{x}_1) \ldots v(\tilde{x}_m))$. A naive proxy is the model's uncertainty about the point.
- The cycle repeats until we collect enough data or the model is good enough.

Stanford
ARTIFICIAL
INTELLIGENCE

1. Introduction to Active Learning



Stanford
ARTIFICIAL
INTELLIGENCE

1. Introduction to Active Learning

- Active learning has been successfully applied to various domains, including computer vision, natural language processing, and recommender systems.
- It is particularly useful when labeling data is expensive or time-consuming.
- Example applications include autonomous driving, medical imaging, meteorology, and fraud detection.

Sanna Jarl, Linus Aronsson, Sadegh Rahrovani, and Morteza Haghir Chehreghani. 2021. "Active Learning of Driving Scenario Trajectories." *Eng. Appl. Artif. Intell.* 113: 104972.

A. Biswas, N. Abdullah Al, M.S. Ali, I. Hossain, M.A. Ullah, S. Talukder (2023). Active Learning on Medical Image. In: Zheng, B., Andrei, S., Sarker, M.K., Gupta, K.D. (eds) *Data Driven Approaches on Medical Imaging*. Springer, Cham.

Aarti Singh, Robert D. Nowak, and Parameswaran Ramanathan. 2006. "Active Learning for Adaptive Mobile Sensing Networks." *2006 5th International Conference on Information Processing in Sensor Networks*, 60–68.

F. Carcillo, YA. Le Borgne, O. Caelen et al. Streaming active learning strategies for real-life credit card fraud detection: assessment and visualization. *Int J Data Sci Anal* 5, 285–300 (2018).

Stanford
ARTIFICIAL
INTELLIGENCE

1. Active Preference Learning

- Active learning can be applied to preference learning tasks, where the goal is to learn a model that aligns with human preferences with limited labeled data and/or high annotation cost.
- For example, we want to teach a robot to cook a meal that you like, but we can only afford to show it a few recipes.
- Active preference learning can help us select the most informative recipes to label and improve the robot's cooking skills efficiently.
- In this section, we will cover the theory behind active preference learning and some examples.

Stanford Artificial Intelligence

2. Uncertainty Quantification

- Uncertainty is an important quantity used in various acquisition functions to qualify the informativeness of a sample. Two types of uncertainty commonly used in ML are:
  - **Epistemic uncertainty** (i.e., model uncertainty) is the uncertainty due to lack of knowledge, which can be reduced with more data.
  - **Aleatoric Uncertainty** (i.e., data uncertainty) is the uncertainty due to inherent randomness in the data that can not be reduced with more data.

Stanford Artificial Intelligence

2. Uncertainty Quantification

There are three common ways to quantify uncertainty in AL:

- **Bayesian Approaches** provide a principled way to quantify uncertainty in models such as Bayesian Neural Networks and Gaussian Processes.
  - *Advantages:* Provide principled uncertainty estimates for various acquisition functions that incorporate prior knowledge.
  - *Disadvantages:* Computationally intractable for many expressive models.

Stanford Artificial Intelligence

2. Uncertainty Quantification

- **Ensemble Approaches** combine multiple models to make predictions. Some common methods are Random Forest and Gradient Boosting.
  - *Advantages:* Easy to implement, can provide uncertainty estimates.
  - *Disadvantages:* Computational expensive, no prior knowledge, may not provide calibrated uncertainty estimates.

Stanford
ARTIFICIAL
INTELLIGENCE

2. Uncertainty Quantification

There are also conformal prediction methods. In this lecture, we will focus on Bayesian approach for qualifying uncertainty.

Stanford
ARTIFICIAL
INTELLIGENCE

3. Acquisition Functions

- Acquisition functions are used to select the most informative samples to label in active learning.
- This function quantifies the utility of labeling a particular sample based on the model's current state.
- Common acquisition functions include uncertainty sampling, query-by-committee, and BALD.

Stanford Artificial Intelligence

3.1. Uncertainty Sampling

- Uncertainty sampling selects samples for which the model is most uncertain. Let x be the input and $p(y|x)$ be the probability of output y given x. Here are some common uncertainty sampling acquisition functions:
  - Entropy Sampling: $\alpha(x) = -\sum_y p(y|x) \log p(y|x)$,
  - Margin Sampling: $\alpha(x) = p(y_1|x) - p(y_2|x)$, where y_1 and y_2 are the two most likely output.
  - Least Confidence Sampling: $\alpha(x) = 1 - p(y_{\mathbf{max}}|x)$, where $y_{\mathbf{max}}$ is the most likely output (output with highest probability).

Zhu, Jingbo, Huizhen Wang, Benjamin Ka-Yin T'sou, and Matthew Y. Ma. 2010. "Active Learning with Sampling by Uncertainty and Density for Data Annotations." *IEEE Transactions on Audio, Speech, and Language Processing* 18: 1323–31.

Stanford
ARTIFICIAL
INTELLIGENCE

3.1. Uncertainty Sampling

Example: Consider a binary classification problem with two classes y_1 and y_2. We have three samples x_1, x_2, x_3 and the corresponding predictive distributions are as follows:

$$
\begin{aligned}
p(y_1|x_1) &= 0.6, \quad p(y_2|x_1) = 0.4 \\
p(y_1|x_2) &= 0.3, \quad p(y_2|x_2) = 0.7 \\
p(y_1|x_3) &= 0.8, \quad p(y_2|x_3) = 0.2
\end{aligned}
$$

Stanford
ARTIFICIAL
INTELLIGENCE

3.1. Uncertainty Sampling

Entropy Sampling

- $\alpha(x_1) = ?$
- $\alpha(x_2) = ?$
- $\alpha(x_3) = ?$

Stanford
ARTIFICIAL
INTELLIGENCE

3.1. Uncertainty Sampling

Entropy Sampling

- $\alpha(x_1) = -0.6\log(0.6) - 0.4\log(0.4) = 0.29$
- $\alpha(x_2) = -0.3\log(0.3) - 0.7\log(0.7) = 0.27$
- $\alpha(x_3) = -0.8\log(0.8) - 0.2\log(0.2) = 0.22$

We would select x_1 for labeling as it has the highest entropy, indicating the model is most uncertain about its prediction at x_1.

Stanford
ARTIFICIAL
INTELLIGENCE

3.1. Uncertainty Sampling

Margin Sampling

- $\alpha(x_1) = ?$
- $\alpha(x_2) = ?$
- $\alpha(x_3) = ?$

Stanford
ARTIFICIAL
INTELLIGENCE

3.1. Uncertainty Sampling

Margin Sampling

- $\alpha(x_1) = 0.6 - 0.4 = 0.2$
- $\alpha(x_2) = 0.7 - 0.3 = 0.4$
- $\alpha(x_3) = 0.8 - 0.2 = 0.6$

We would select x_1 for labeling as it has the smallest margin, indicating the model is most uncertain about the prediction at x_1.

Stanford
ARTIFICIAL
INTELLIGENCE

3.1. Uncertainty Sampling

Least Confidence Sampling

- $\alpha(x_1) = ?$
- $\alpha(x_2) = ?$
- $\alpha(x_3) = ?$

Stanford
ARTIFICIAL
INTELLIGENCE

3.1. Uncertainty Sampling

Least Confidence Sampling

- $\alpha(x_1) = 1 - 0.6 = 0.4$
- $\alpha(x_2) = 1 - 0.7 = 0.3$
- $\alpha(x_3) = 1 - 0.8 = 0.2$

We would select x_1 for labeling as it has the lowest confidence, indicating the model is most uncertain about the prediction at x_1.

Stanford
ARTIFICIAL
INTELLIGENCE

3.2. Query-by-Committee

- The Query-by-Committee selects samples for which the committee members disagree the most. Given C committee members and $\mathbb{H}$ is the entropy function.
  - Vote Entropy: $\alpha(x) = \mathbb{H}\left[\frac{V(y)}{C}\right]$, where $V(y)$ is the number of votes for class y.
  - Consensus Entropy: $\alpha(x) = \mathbb{H}[P_C(y|x)]$ where $P_C(y|x)$ is the average probability distribution of the committee members.
  - KL Divergence: $\alpha(x) = \frac{1}{C}\sum_{c=1}^{C} D_{KL}[P_c(y|x)||P_C(y|x)]$.

Zhu, Jingbo, Huizhen Wang, Benjamin Ka-Yin T'sou, and Matthew Y. Ma. 2010. "Active Learning with Sampling by Uncertainty and Density for Data Annotations." *IEEE Transactions on Audio, Speech, and Language Processing* 18: 1323–31.

Stanford
ARTIFICIAL
INTELLIGENCE

3.2. Query-by-Committee

Example: Consider a binary classification problem with two classes y_1 and y_2. We have three committee members and three samples x_1, x_2, x_3. The committee members' predictive distributions are as follows, where $p_i(y_j|x_k)$ is the probability of committee member i predicting class y_j given input x_k.

x	$p_1(y_1\|\cdot)$	$p_1(y_2\|\cdot)$	$p_2(y_1\|\cdot)$	$p_2(y_2\|\cdot)$	$p_3(y_1\|\cdot)$	$p_3(y_2\|\cdot)$
x_1	0.6	0.4	0.7	0.3	0.3	0.7
x_2	0.3	0.7	0.4	0.6	0.4	0.6
x_3	0.8	0.2	0.9	0.1	0.7	0.3

Stanford Artificial Intelligence

3.2. Query-by-Committee

Vote Entropy for x_1

- Vote for y_1: $V(y_1) = ?$
- Vote for y_2: $V(y_2) = ?$
- $\alpha(x_1) = ?$

Vote Entropy for x_2

- Vote for y_1: $V(y_1) = ?$
- Vote for y_2: $V(y_2) = ?$
- $\alpha(x_2) = ?$

Vote Entropy for x_3

- Vote for y_1: $V(y_1) = ?$
- Vote for y_2: $V(y_2) = ?$
- $\alpha(x_3) = ?$

Stanford Artificial Intelligence

3.2. Query-by-Committee

Vote Entropy for x_1

- Vote for y_1: $V(y_1) = 2$
- Vote for y_2: $V(y_2) = 1$
- $\alpha(x_1) = -\frac{2}{3}\log(\frac{2}{3}) - \frac{1}{3}\log(\frac{1}{3}) = 0.28$

Vote Entropy for x_2

- Vote for y_1: $V(y_1) = 0$
- Vote for y_2: $V(y_2) = 3$
- $\alpha(x_2) = -\frac{0}{3}\log(\frac{0}{3}) - \frac{3}{3}\log(\frac{3}{3}) = 0$

Vote Entropy for x_3

- Vote for y_1: $V(y_1) = 3$
- Vote for y_2: $V(y_2) = 0$
- $\alpha(x_3) = -\frac{3}{3}\log(\frac{3}{3}) - \frac{0}{3}\log(\frac{0}{3}) = 0$

Stanford
ARTIFICIAL
INTELLIGENCE

3.2. Query-by-Committee

Vote Entropy

- $\alpha(x_1) = 0.28$
- $\alpha(x_2) = 0$
- $\alpha(x_3) = 0$

Thus, we would select x_1 for labeling as it has the highest vote entropy, indicating the committee members disagree the most about the prediction at x_1.

Stanford Artificial Intelligence

3.2. Query-by-Committee

Consensus Entropy

- **Step 1:** Compute the consensus probability of each class for each sample.
  - $p_c(y_1|x_1) = ?$
  - $p_c(y_2|x_1) = ?$
  - $p_c(y_1|x_2) = ?$
  - $p_c(y_2|x_2) = ?$
  - $p_c(y_1|x_3) = ?$
  - $p_c(y_2|x_3) = ?$
- **Step 2:** Compute the entropy of the consensus probability for each sample.
  - $\mathbb{H}[p_c(y|x_1)] = ?$
  - $\mathbb{H}[p_c(y|x_2)] = ?$
  - $\mathbb{H}[p_c(y|x_3)] = ?$

Stanford
ARTIFICIAL
INTELLIGENCE

3.2. Query-by-Committee

Consensus Entropy

- **Step 1:** Compute the consensus probability of each class for each sample.
  - $p_c(y_1|x_1) = \frac{0.6+0.7+0.3}{3} = 0.53$
  - $p_c(y_2|x_1) = \frac{0.4+0.3+0.7}{3} = 0.47$
  - $p_c(y_1|x_2) = \frac{0.3+0.4+0.4}{3} = 0.37$
  - $p_c(y_2|x_2) = \frac{0.7+0.6+0.6}{3} = 0.63$
  - $p_c(y_1|x_3) = \frac{0.8+0.9+0.7}{3} = 0.8$
  - $p_c(y_2|x_3) = \frac{0.2+0.1+0.3}{3} = 0.2$

Stanford
ARTIFICIAL
INTELLIGENCE

3.2. Query-by-Committee

Consensus Entropy

- **Step 2:** Compute the entropy of the consensus probability for each sample.
  - $\mathbb{H}[p_c(y|x_1)] = -0.53\log(0.53) - 0.47\log(0.47) = 0.30$
  - $\mathbb{H}[p_c(y|x_2)] = -0.37\log(0.37) - 0.63\log(0.63) = 0.29$
  - $\mathbb{H}[p_c(y|x_3)] = -0.8\log(0.8) - 0.2\log(0.2) = 0.22$

We would select x_1 for labeling as it has the highest consensus entropy, indicating the committee members disagree the most about the prediction at x_1.

Stanford Artificial Intelligence

3.3. Bayesian Active Learning by Disagreement

Bayesian Active Learning by Disagreement (BALD) selects the samples for which the model believes the most (Shannon) information can be gained in expectation if these corresponding labels are observed:

$$\mathbb{I}(\theta; y|x, \mathcal{D}) = \mathbb{H}[p(y|x, \mathcal{D})] - \mathbb{E}_{p(\theta|\mathcal{D})}[\mathbb{H}[p(y|x, \theta, \mathcal{D})]]$$

where $\mathbb{H}[\cdot]$ denotes entropy. When there is significant disagreement among models, the predictive entropy (first term) will be large, but the expected entropy (second term)) will be lower. This difference represents how much the models disagree with each other. BALD selects points where this disagreement is maximized.

Houlsby, N., Huszár, F., Ghahramani, Z., & Lengyel, M. (2011). Bayesian Active Learning for Classification and Preference Learning. ArXiv, abs/1112.5745.

Stanford
ARTIFICIAL
INTELLIGENCE

3.3. Bayesian Active Learning by Disagreement

To compute the first term, we can derive the following expression:

$$\begin{aligned}\mathbb{H}[p(y|x,\mathcal{D})] &= \mathbb{H}\left[\int_\theta p(y|x,\theta,\mathcal{D})p(\theta|\mathcal{D})d\theta\right] \\ &\approx \mathbb{H}\left[\frac{1}{N}\sum_{i=1}^{N} p(y|x,\theta_i,\mathcal{D})\right] \\ &= \mathbb{H}\left[\overline{p}(y|x,\mathcal{D})\right]\end{aligned}$$

Stanford
ARTIFICIAL
INTELLIGENCE

3.3. Bayesian Active Learning by Disagreement

To compute the second term, we can derive the following expression:

$$
\begin{aligned}
&\mathbb{E}_{p(\theta|\mathcal{D})}[\mathbb{H}[p(y|x,\theta,\mathcal{D})]] \\
&= \mathbb{E}_{p(\theta|\mathcal{D})}\left[-\sum_y p(y|x,\theta,\mathcal{D}) \log p(y|x,\theta,\mathcal{D})\right] \\
&\approx -\frac{1}{N}\sum_{i=1}^{N}\left(\sum_y p(y|x,\theta_i,\mathcal{D}) \log p(y|x,\theta_i,\mathcal{D})\right)
\end{aligned}
$$

Stanford ARTIFICIAL INTELLIGENCE

3.3. Bayesian Active Learning by Disagreement

Example: Consider a binary classification problem with two classes y_1 and y_2. We have two samples x_1, x_2 and the model's predictive distributions are as follows:

First-time inference (with $\theta_1 \sim p(\theta|\mathcal{D})$)

$$p(y_1|x_1, \theta_1, \mathcal{D}) = 0.6, \quad p(y_2|x_1, \theta_1, \mathcal{D}) = 0.4$$
$$p(y_1|x_2, \theta_1, \mathcal{D}) = 0.4, \quad p(y_2|x_2, \theta_1, \mathcal{D}) = 0.6$$

Second-time inference (with $\theta_2 \sim p(\theta|\mathcal{D})$)

$$p(y_1|x_1, \theta_2, \mathcal{D}) = 0.8, \quad p(y_2|x_1, \theta_2, \mathcal{D}) = 0.2$$
$$p(y_1|x_2, \theta_2, \mathcal{D}) = 0.5, \quad p(y_2|x_2, \theta_2, \mathcal{D}) = 0.5$$

Stanford
ARTIFICIAL
INTELLIGENCE

3.3. Bayesian Active Learning by Disagreement

Step 1: Compute the entropy of the model's predictive distribution for each sample.

- $\mathbb{H}[p(y|x_1, \mathcal{D})] = ?$
- $\mathbb{H}[p(y|x_2, \mathcal{D})] = ?$

Step 2: Compute the expected entropy of the model's predictive distribution for each sample.

- $\mathbb{E}_{p(\theta|\mathcal{D})}[\mathbb{H}[p(y|x_1, \theta, \mathcal{D})]] = ?$
- $\mathbb{E}_{p(\theta|\mathcal{D})}[\mathbb{H}[p(y|x_2, \theta, \mathcal{D})]] = ?$

Stanford
ARTIFICIAL
INTELLIGENCE

3.3. Bayesian Active Learning by Disagreement

Step 1: Compute the entropy of the model's predictive distribution for each sample.

- $\overline{p}_\theta(y_1|x_1, \theta, \mathcal{D}) = 0.7$
- $\overline{p}_\theta(y_2|x_1, \theta, \mathcal{D}) = 0.3$
- $\overline{p}_\theta(y_1|x_2, \theta, \mathcal{D}) = 0.45$
- $\overline{p}_\theta(y_2|x_2, \theta, \mathcal{D}) = 0.55$
- $\mathbb{H}[p(y|x_1, \mathcal{D})] = -0.7\log(0.7) - 0.3\log(0.3) = 0.27$
- $\mathbb{H}[p(y|x_2, \mathcal{D})] = -0.45\log(0.45) - 0.55\log(0.55) = 0.30$

Stanford
ARTIFICIAL
INTELLIGENCE

3.3. Bayesian Active Learning by Disagreement

Step 2: Compute the expected entropy of the model's predictive distribution for each sample.

- $\mathbb{H}_{\theta_1}[p(y|x_1, \theta, \mathcal{D})] = -0.6\log(0.6) - 0.4\log(0.4) = 0.29$
- $\mathbb{H}_{\theta_2}[p(y|x_1, \theta, \mathcal{D})] = -0.8\log(0.8) - 0.2\log(0.2) = 0.22$

$$\Rightarrow \mathbb{E}_{p(\theta|\mathcal{D})}[\mathbb{H}[p(y|x_1, \theta, \mathcal{D})]] \approx (0.29 + 0.22)/2 = 0.255$$

- $\mathbb{H}_{\theta_1}[p(y|x_2, \theta, \mathcal{D})] = -0.4\log(0.4) - 0.6\log(0.6) = 0.29$
- $\mathbb{H}_{\theta_2}[p(y|x_2, \theta, \mathcal{D})] = -0.5\log(0.5) - 0.5\log(0.5) = 0.30$

$$\Rightarrow \mathbb{E}_{p(\theta|\mathcal{D})}[\mathbb{H}[p(y|x_2, \theta, \mathcal{D})]] \approx (0.29 + 0.30)/2 = 0.295$$

Stanford
ARTIFICIAL
INTELLIGENCE

3.3. Bayesian Active Learning by Disagreement

Step 3: Compute the BALD score for each sample.

- $\alpha(x_1) = 0.27 - 0.255 = 0.015$
- $\alpha(x_2) = 0.30 - 0.295 = 0.005$

We would select x_1 for labeling as it has the highest BALD score, indicating the model will gain the most information from labeling x_1.

Stanford
ARTIFICIAL
INTELLIGENCE

3.4 Active Preference Learning by Variance Reduction

- In this method, the main idea is to select new point $\tilde{x} \sim p(x)$ for which the model believes $P(\tilde{Y}|\tilde{X} = \tilde{x})$ if labeled as $y(\tilde{x})$ and added to $\mathcal{D}$, will reduce the variance of the model the most.

David A. Cohn, Zoubin Ghahramani, and Michael I. Jordan. 1996. "Active Learning with Statistical Models." CoRR cs.AI/9603104.

3.4 Active Preference Learning by Variance Reduction

- Starting from the expected error at $\boldsymbol{x}$, we have:

$$\mathbb{E}_{\hat{y}\sim p(\hat{y}|\mathcal{D};x),y\sim p(y|x)}(\hat{y}-y)^2$$

where $\hat{y}$ is the model prediction, y is the true label.

- Geman et al., 1992:

$$\begin{aligned}\mathbb{E}_{\hat{y}\sim p(\hat{y}|\mathcal{D};x),y\sim p(y|x)}(\hat{y}-y)^2 &= \mathbb{E}_{\hat{y},y}[(\hat{y}-\mathbb{E}[y|x])+(\mathbb{E}[y|x]-y)]^2 \\ &= \mathbb{E}_{\hat{y},y}[(y-\mathbb{E}[y|x])^2] \\ &+ 2\mathbb{E}_{\hat{y},y}[(\hat{y}-\mathbb{E}[y|x])(\mathbb{E}[y|x]-y)] \\ &+ \mathbb{E}_{\hat{y},y}(\hat{y}-\mathbb{E}[y|x])^2\end{aligned}$$

Stuart Geman, Elie Bienenstock, and René Doursat. 1992. "Neural Networks and the Bias/Variance Dilemma." Neural Computation 4: 1–58.

3.4 Active Preference Learning by Variance Reduction

- The second term $2\mathbb{E}_{\hat{y},y}[(\hat{y} - \mathbb{E}[y|x])(\mathbb{E}[y|x] - y)]$ is zero as

$$\mathbb{E}_{\hat{y},y}[\mathbb{E}[y|x] - y] = \mathbb{E}_y[\mathbb{E}[y|x]] - \mathbb{E}_y[y] = \mathbb{E}_y[y] - \mathbb{E}_y[y] = 0$$

- Continue to derive the third term:

$$\begin{aligned}
\mathbb{E}_{\hat{y},y}(\hat{y} - \mathbb{E}[y|x])^2 &= \mathbb{E}_{\hat{y},y}[(\hat{y} - \mathbb{E}_{\hat{y}}[\hat{y}] + \mathbb{E}_{\hat{y}}[\hat{y}] - \mathbb{E}[y|x])^2] \\
&= \mathbb{E}_{\hat{y},y}[(\hat{y} - \mathbb{E}_{\hat{y}}[\hat{y}])^2] \\
&+ 2\mathbb{E}_{\hat{y},y}\left[(\hat{y} - \mathbb{E}_{\hat{y}}[\hat{y}])(\mathbb{E}_{\hat{y}}[\hat{y}] - \mathbb{E}[y|x])\right] \\
&+ \mathbb{E}_{\hat{y},y}[(\mathbb{E}_{\hat{y}}[\hat{y}] - \mathbb{E}[y|x])^2] \\
&= \mathbb{E}_{\hat{y}}[(\hat{y} - \mathbb{E}_{\hat{y}}[\hat{y}])^2] + (\mathbb{E}_{\hat{y}}[\hat{y}] - \mathbb{E}[y|x])^2
\end{aligned}$$

as $\mathbb{E}_{\hat{y},y}(\hat{y} - \mathbb{E}_{\hat{y}}[\hat{y}]) = \mathbb{E}_{\hat{y}}[\hat{y} - \mathbb{E}_{\hat{y}}[\hat{y}]] = \mathbb{E}_{\hat{y}}[\hat{y}] - \mathbb{E}_{\hat{y}}[\hat{y}] = 0$,
$\mathbb{E}_{\hat{y}}[\hat{y}] = \mathbb{E}_{\hat{y} \sim p(\hat{y}|\mathcal{D},x)}[\hat{y}|x,\mathcal{D}]$, and $p(\hat{y}|\mathcal{D},x)$ is the posterior predictive.

Stanford Artificial Intelligence

3.4 Active Preference Learning by Variance Reduction

Finally, we have:

$$\mathbb{E}_y[(y - \mathbb{E}[y|x])^2] + (\mathbb{E}_{\hat{y}}[\hat{y} - \mathbb{E}[y|x]])^2 + \mathbb{E}_{\hat{y}}[(\hat{y} - \mathbb{E}_{\hat{y}}[\hat{y}])^2]$$

- The first term is the variance of the true label, which is independent of the model. We can not change it.
- The second term is the bias of the model. We do not have control over it when selecting data.
- The third term is the variance of the model prediction given the previous data $\mathcal{D}$, which can be used to quantify the model's uncertainty about the chosen x.

Stuart Geman, Elie Bienenstock, and René Doursat. 1992. "Neural Networks and the Bias/Variance Dilemma." Neural Computation 4: 1–58.

Stanford
ARTIFICIAL
INTELLIGENCE

3.4 Active Preference Learning by Variance Reduction

- Following Cohn et al. (1996), we can denote the third term as:

$$\sigma^2_{\hat{y}}(x|\mathcal{D}) = \mathbb{E}_{\hat{y}}[(\hat{y} - \mathbb{E}_{\hat{y}}[\hat{y}])^2]$$

- Written it more explicitly:

$$\sigma^2_{\hat{y}}(x|\mathcal{D}) = \mathbb{E}_{\hat{y}\sim p(\hat{y}|\mathcal{D};x)}[(\hat{y} - \mathbb{E}_{\hat{y}\sim p(\hat{y}|\mathcal{D};x)}[\hat{y}])^2]$$

David A. Cohn, Zoubin Ghahramani, and Michael I. Jordan. 1996. “Active Learning with Statistical Models.” CoRR cs.AI/9603104.

Stanford Artificial Intelligence

3.4 Active Preference Learning by Variance Reduction

Recall $x \sim p(x)$ is a sample from the input distribution and $\tilde{x}$ is a candidate from the active learning pool. We define $\tilde{\mathcal{D}} = \mathcal{D} \cup \{(\tilde{x}, y(\tilde{x}))\}$. The active learning procedure is as follows:

1. Sample candidate points $\tilde{x}_1, \ldots, \tilde{x}_m$ from $p(x)$.
2. For each candidate point $\tilde{x}_i$, compute the expected variance reduction

$$\mathbb{E}_{p(x)}[\sigma^2_{\hat{y}}(x|\tilde{\mathcal{D}})]$$

3. Select the point $\tilde{x}^*$ to label.

$$\tilde{x}^* = \arg\min_{\tilde{x}_i} \mathbb{E}_{p(x)}[\sigma^2_{\hat{y}}(x|\tilde{\mathcal{D}})]$$

4. Update the model with newly observed data and repeat.

Stanford Artificial Intelligence

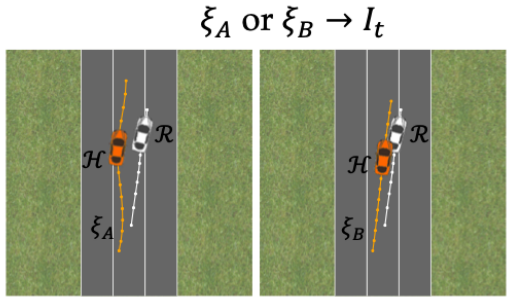
3.5. Other Acquisition Functions

- There are many other acquisition functions that can be used in active learning, including:
  - Expected Model Change (Cai et al., 2013)
  - Expected Error Reduction (Mussmann et al., 2022)
  - Variance Reduction (Cohn et al., 1996)
  - Active Thompson Sampling (Bouneffouf et al., 2014)
  - Mismatch-first Farthest-traversal (Zhao et al., 2020)

Wenbin Cai, Ya Zhang, and Jun Zhou. 2013. "Maximizing Expected Model Change for Active Learning in Regression." In *2013 IEEE 13th International Conference on Data Mining*, 51–60.
Stephen Mussmann, Julia Reisler, Daniel Tsai, Ehsan Mousavi, Shayne O'Brien, and Moises Goldszmidt. 2022. "Active Learning with Expected Error Reduction."
David A. Cohn, Zoubin Ghahramani, and Michael I. Jordan. 1996. "Active Learning with Statistical Models." CoRR cs.AI/9603104.
Djallel Bouneffouf, Romain Laroche, Tanguy Urvoy, Raphaël Féraud, and Robin Allesiardo. 2014. "Contextual Bandit for Active Learning: Active Thompson Sampling." In *International Conference on Neural Information Processing*.
Shuyang Zhao, Toni Heittola, and Tuomas Virtanen. 2020. "Active Learning for Sound Event Detection." *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 28: 2895–905.

4. Case study: Active Preference-based Learning of Reward Functions

Example in autonomous driving: Two candidate trajectories are provided for comparison. We can observe that ζ_A has a smoother trajectory without any collisions.



- Active preference-based learning can be used to learn reward functions in reinforcement learning. It is used to answer the following question:

```
What if humans do not precisely know how an agent should optimally behave in
an environment but still have some opinion on what trajectories would be
better than others?
```

Stanford Artificial Intelligence

4. Case study: Active Preference-based Learning of Reward Functions

- Let us consider a scenario where a **human** expert provides controls on the **robot**'s behavior by comparing two trajectories.
- Let a state at step t be x_t, u_H be the human prefered control, and u_R be the robot's control.
- We define the next state as $x^{t+1} = f_{HR}(x^t, u_R^t, u_H^t)$.
- We can model the human's preference as a reward function

$$r_H(x^t, u_R^t, u_H^t) = \mathbf{w}^\top \phi(x_t, u_R^t, u_H^t)$$

where $\mathbf{w}$ is the weight vector, ϕ is the feature function.

Dorsa Sadigh et al. Active preference-based learning of reward functions. 2017.

Stanford Artificial Intelligence

4. Case study: Active Preference-based Learning of Reward Functions

- The goal is to learn the expected reward function R_H over horizon N from the human's preferences.

$$R_H(x^0, \mathbf{u}_R, \mathbf{u}_H) = \sum_{t=0}^{N} r_H(x^t, u_R^t, u_H^t)$$

- We can define a controled trajectory $\zeta \in \Xi$ as $\zeta = (x^0, u_R^0, u_H^0), \ldots, (x^N, u_R^N, u_H^N)$ and $\Phi(\zeta) = \sum_{t=0}^{N} \phi(x^t, u_R^t, u_H^t)$.
- Thus, we can rewrite the expected reward function as

$$R_H(\zeta) = \mathbf{w}^\top \Phi(\zeta)$$

Dorsa Sadigh et al. Active preference-based learning of reward functions. 2017.

Stanford
ARTIFICIAL
INTELLIGENCE

4. Case study: Active Preference-based Learning of Reward Functions

Recall the previous lecture on preference learning, we can define the probability of the human preferring trajectory ζ_A over ζ_B as

$$
\begin{aligned}
P(\zeta_A \succ \zeta_B) &= \sigma(R_H(\zeta_A) - R_H(\zeta_B)) \\
&= \frac{1}{1 + \exp(-(R_H(\zeta_A) - R_H(\zeta_B)))} \\
&= \frac{\exp(R_H(\zeta_A))}{\exp(R_H(\zeta_A)) + \exp(R_H(\zeta_B))}
\end{aligned}
$$

Do you find the above equation familiar?

Stanford Artificial Intelligence

4. Case study: Active Preference-based Learning of Reward Functions

With $R_H(\zeta)$, we can define the probability of the human preferring trajectory ζ_A over ζ_B as

$$p(I = 1|\mathbf{w}) = \frac{\exp(R_H(\zeta_A))}{\exp(R_H(\zeta_A)) + \exp(R_H(\zeta_B))}$$

and the probability of the human preferring trajectory ζ_B over ζ_A as

$$p(I = -1|\mathbf{w}) = \frac{\exp(R_H(\zeta_B))}{\exp(R_H(\zeta_A)) + \exp(R_H(\zeta_B))}$$

Dorsa Sadigh et al. Active preference-based learning of reward functions. 2017.

Stanford
ARTIFICIAL
INTELLIGENCE

4. Case study: Active Preference-based Learning of Reward Functions

We can rewrite the probability of the human preferring trajectory ζ_A over ζ_B with $\psi = \Phi(\zeta_A) - \Phi(\zeta_B)$, $\Phi(\zeta) = \sum_0^N (x^t, u_R^t, u_H^t)$ as

$$f_\psi(\mathbf{w}) = P(I|\mathbf{w}) = \frac{1}{1 + \exp(-I\mathbf{w}^\top \psi)}$$

The idea is that we can use a Bayesian update to compute the posterior of the weight vector $\mathbf{w}$ given the human's preferences.

$$p(\mathbf{w}|I) \propto p(I|\mathbf{w})p(\mathbf{w})$$

Dorsa Sadigh et al. Active preference-based learning of reward functions. 2017.

Stanford
ARTIFICIAL
INTELLIGENCE

4. Case study: Active Preference-based Learning of Reward Functions

The acquisition function used in this case is the following:

$$\max_{\psi} \{\min\{\mathbb{E}[1 - f_{\psi}(\mathbf{w})], \mathbb{E}[1 - f_{-\psi}(\mathbf{w})]\}\}$$

Instead of optimizing on ψ, we can reformulate the optimization problem as

$$\max_{x^0, \mathbf{u}_R, \mathbf{u}_H^A, \mathbf{u}_H^B} \{\min\{\mathbb{E}[1 - f_{\psi}(\mathbf{w})], \mathbb{E}[1 - f_{-\psi}(\mathbf{w})]\}\}$$

Dorsa Sadigh et al. Active preference-based learning of reward functions. 2017.

Stanford Artificial Intelligence

4. Case study: Active Preference-based Learning of Reward Functions

To solve this optimization problem, in case $p(\mathbf{w})$ is complex, we can approximate this $p(\mathbf{w})$ by the empirical distribution of the weight vector $\mathbf{w}$.

$$p(\mathbf{w}) \approx \frac{1}{M} \sum_{i=1}^{M} \delta(\mathbf{w}_i)$$

where δ is the Dirac delta function. Then, we can rewrite the optimization problem as

$$\max_{x^0, \mathbf{u}_R, \mathbf{u}_H^A, \mathbf{u}_H^B} \left\{ \min \left\{ \frac{1}{M} \sum_{i=1}^{M} 1 - f_{\psi}(\mathbf{w}_i), \frac{1}{M} \sum_{i=1}^{M} 1 - f_{-\psi}(\mathbf{w}_i) \right\} \right\}$$

Now, we can use gradient-based optimization methods to solve this optimization problem (e.g. Quasi-Newton methods).

Stanford Artificial Intelligence

Discussion and QA

Summary

- Active learning is a machine learning paradigm that aims to reduce the amount of labeled data required to train a model.
- Active preference learning is used to learn models that align with human preferences with limited labeled data and/or high annotation cost.
- Acquisition functions are used to select the most informative samples to label in active learning.

Stanford
ARTIFICIAL
INTELLIGENCE

Discussion and QA

Discussion

- What are some challenges of active preference learning? How can we address them? E.g., scalability, human feedback quality, etc.
- Active preference learning can be used in various applications. Can you think of other applications where active preference learning can be useful?

Next lecture: Model-free Preference Optimization