CS329X: Human Centered NLP

Human in the Loop

Diyi Yang
Stanford CS
Overview

Reasons we need human feedback

How models can take feedback

How humans can give feedback

Many slides credit to Sherry Wu
Why do we need human feedback?

A misalignment between this fine-tuning objective (maximizing the likelihood of human-written text) and what we care about (generating high-quality outputs as determined by humans).

The objective function mixes important errors (making up facts) and unimportant errors (selecting the precise word from a set of synonyms)

Models are incentivized to place probability mass on all human demonstrations, including those that are low-quality.
Some common objectives for human feedback...

A misalignment between this fine-tuning objective (maximizing the likelihood of human-written text) and what we care about (generating high-quality outputs as determined by humans).

Make model output more aligned with our values:

Model performance, robustness and generalizability (aligned with our expectations on model behaviors)

Fairness (aligned with our societal values)

Explainability (aligned with our rationales)

Personal beliefs

What feedback can you imagine giving to a model?

Many forms, but might depends on what the model can take!
Keys of Human-in-the-loop NLP

Allow humans to **easily provide feedback**.

Turn **nontechnical, human preferences** into **usable model updates**.

Build models to **effectively take the feedback**.

Human in the loop NLP has a “long” history

Interactive Text Classification

Document-level interaction
- Suggest documents to label
- Suggest labels for documents
- Check label consistency

Word-level interaction
- Suggest influential terms
- Accept engineered features
- Add/drop features

Model+data exploration
- Term by class parameters
- Drill down into documents
- Accuracy summaries

Human in the loop NLP has a “long” history

Pat ate the cake on the table that I *baked* last night.

Parser: I *baked* table
Human understanding: I *baked* cake

Human-in-the-Loop Parsing

Luheng He, Julian Michael, *Mike Lewis, Luke Zettlemoyer
University of Washington

Human in the loop NLP has a “long” history

Candidate dependencies from the n-best list:
- baked → table
- baked → cake

Q: “What did someone bake?”
1) table
2) cake

Re-parsed CCG Dependency Tree

Not re-training the model

Human-in-the-Loop Parsing

Luheng He, Julian Michael, *Mike Lewis, Luke Zettlemoyer
University of Washington
Human in the loop NLP has a “long” history

Interactive Topic Modeling: start with a vanilla LDA with symmetric prior, get the initial topics. Then repeat the following process till users are satisfied: show users topics, get feedback from users, encode the feedback into a tree prior, update topics with tree-based LDA.
User interface for the HL-TM tool. A list of topics (left) are represented by topics’ first three topic words. Selecting a topic reveals more detail (right): the top 20 words and top 40 documents. Hovering or clicking on a word highlights it within the documents. Users can refine the model using simple mechanisms: click “x” next to words or documents to remove them, select and drag words to re-order them, type new words from the vocabulary into the input box and press “enter” to add them, select a word and click the trash can to add it to the stop words list, or click “split” and “merge” (to the right of the topic words) to enter into split and merge modes.
Keys of Human-in-the-loop NLP

Allow humans to **easily provide feedback**.

Turn **nontechnical, human preferences** into **usable model updates**.

**Build models to effectively take the feedback.**

Taxonomy: Levels of domain expert feedback

**Observation-level feedback** (local)
Infer preferences from human judgements on each data points
e.g., radiologist provide gold annotations on X-ray scans

**Domain-level feedback** (global)
Provide explicit feedback on the entire task.
e.g., radiologist provide high-level descriptions about the region of interest in X-Rays
Taxonomy: Types of Model Updates

The supervised learning setting

\[ \hat{\theta} = \arg\min_{\theta \in \Theta} \sum_{(x, y) \in D} L(x, y; \theta) \]

Learn a model parametrized by \( \theta \in \Theta \)

By minimizing a objective function

On a dataset \( D \)
Taxonomy: Types of Model Updates

\[ \hat{\theta} = \arg \min_{\theta \in \Theta} \sum_{(x, y) \in D} L(x, y; \theta) \]

**Dataset updates.** change the dataset  
* e.g., add / remove appropriate datapoints

**Loss function updates.** add a constraint to the optimization objective  
* e.g., add a regularizer that penalizes the model for not satisfying this condition

**Parameter space updates.** Change the model parameters  
* e.g., optimize over a subspace of parameters
Update Datasets (aka Data curation)

Global: systematically add data points
Data augmentation
Resampling

Local: Iteratively add data points
Active learning
model-assisted adversarial labeling
Global data update: Weak supervision

Weak supervision: Use imperfect or noisy sources of supervision to train models.

Snorkel key idea: data is key, but data collection is too expensive.

We should try using noisy sources of signal, specified at higher-levels of abstraction, to rapidly generate training sets.

"We find that Chemical A likely does not cause Disease X."

```python
def labeling_function_1(x):
    if re.find(r'not', x.between):
        return False
```


Slides adjusted from Alex Ratner’s presentation
Global data update: Weak supervision

Input: LFs, Unlabeled data
Use LFs to produce Noisy, conflicting labels

```python
def LF_pneumothorax(c):
    if re.search(r'pneumo.*', c.report.text):
        return "ABNORMAL"

def LF_pleral_effusion(c):
    if "pleural effusion" in c.report.text:
        return "ABNORMAL"
```

Global data update: Weak supervision

Input: LFs, Unlabeled data

Use LFs to produce Noisy, conflicting labels

Label Model

Resolve conflicts, re-weight & combine

Learned Accuracies

Probabilistic Training Labels

90%  

80%  

60%

LF 1  

LF 2  

LF 3  

$X_1$  

$X_2$  

$X_3$
Global data update: Weak supervision

Input: LFs, Unlabeled data
- Use LFs to produce Noisy, conflicting labels

Label Model
- Resolve conflicts, re-weight & combine

End Model
- Generalize beyond labeling functions
Local data update: Active learning

**Active learning**: Proactively select which data points we want to use to learn from, rather than passively accepting all data points available.

**Intuition**: If we have limited labeling budget, some data points are more useful for learning the true decision boundary than others.
Local data update: Active learning

**Active learning**: Proactively select which data points we want to use to learn from, rather than passively accepting all data points available.

- **400 instances sampled**
- **random sampling**: 30 labeled instances (accuracy=0.7)
- **uncertainty sampling**: 30 labeled instances (accuracy=0.9)

There are multiple ways to estimate “usefulness”, e.g. **uncertainty**.
We provide this form of feedback...

Mostly at places where we have data.

Local vs. Global feedback

As you will also see in other examples...

Global feedback tends to be

- **More explicit.** requires you to specify what you want
- **More “intrusive”** & has **larger impacts.** e.g., you can use LF on 10k+ data

Be cautious about making large but not thoughtful changes!

Local feedback tends to be

- **More implicit.** Goals are inferred – which means can be wrong!
- **Less impactful.** Goals are inferred from a set of smaller tweaks, e.g., you only label 100 examples in active earning!

Be cautious about making too trivial or counter-intuitive tweaks!
Update Loss Function (aka model regularization)

Basically, change the way model is optimized, by adding constraints to the optimization objective.

**Global:** Explicitly add regularization to specifies model behavior,

**Local:** infer constraints from expert feedback on individual points (e.g. yellow is a more severe error)
Global loss func update: Unlikelihood training

Penalize undesirable generations (e.g. not following control, repeating previous context)

\[
\mathcal{L}_{MLE} \quad \text{... starboard engines and was going to crash. “We’re going in,” he said. “We’re going to crash. We’re going to crash. We’re going to crash. We’re going to crash. We’re going to crash. We’re going to Hood said. “I’m going to make sure we’re going to get back to the water.” The order to abandon ship was given by Admiral Beatty, who ordered the remaining two battlecruisers to turn away. At 18:25, Hood turned his}
\]

\[
\mathcal{L}_{ULE}^t = \mathcal{L}_{MLE}^t + \alpha \mathcal{L}_{UL}^t
\]

General language model training objective

Another objective that lower the likelihood of undesired tokens \(C\)

\[
\mathcal{L}_{UL}^t = - \sum_{y_{neg} \in C} \log(1 - P(y_{neg} \mid \{y^*\}_{<t}))
\]

e.g. if \(C\) is previously seen text, then less repetition and more diversity

We provide this form of feedback...

Mostly when we decide what model structure to use.

Update Parameter Space (aka model editing)

Basically, directly change the parameters in the model so it uses the information in each data point differently from when it’s unedited.

Global: Explicitly edit model parameters

Local: change the feature space (then the weights of those features become 0)
Train model to explicitly use human-provided concepts.

Global Parameter Space update: Concept Bottleneck Model

Concept bottlenecks enable interventions.

Global Parameter Space update: Concept Bottleneck Model

Concept bottlenecks enable interventions.

Global Parameter Space update: Concept Bottleneck Model

Concept bottlenecks enable interventions.

Local Parameter Space Update: feature engineering/patching

Dynamically fix model bugs by specifying feature/label space using natural language patches.

Note that this is a more explicit form of local feedback!

We provide this form of feedback...

During and after the model is trained.

Reinforcement Learning from Human Feedback

- Output of step 1
  - Initial Language Model
    - Base Text
      - y: a furry mammal

- Output of step 2
  - Tuned Language Model (RL Policy)
    - Parameters Frozen
    - RLHF Tuned Text
      - y: man's best friend

- Output of step 3
  - Reward (Preference) Model
    - text
    - $r^g$

- KL prediction shift penalty
  - $-\lambda_{KL} D_{KL} \left( \pi_{PPO} (y|\epsilon) \| \pi_{base} (y|\epsilon) \right)$

Nathan Lambert: Intro to Reinforcement Learning from Human Feedback
# Feedback-Update Taxonomy

<table>
<thead>
<tr>
<th>Domain</th>
<th>Dataset Update</th>
<th>Loss Function Update</th>
<th>Parameter Space Update</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dataset modification</td>
<td>Augmentation Preprocessing</td>
<td>Constraint specification</td>
<td>Model editing</td>
</tr>
<tr>
<td>Data generation from constraint</td>
<td>Fairness, weak supervision</td>
<td>Fairness, Interpretability</td>
<td>Rules, Weights</td>
</tr>
<tr>
<td>Use unlabeled data</td>
<td></td>
<td>Resource constraints</td>
<td>Model selection</td>
</tr>
<tr>
<td>Check synthetic data</td>
<td></td>
<td></td>
<td>Prior update, Complexity</td>
</tr>
<tr>
<td>Active data collection</td>
<td>Add data, Relabel data, Reweight data</td>
<td>Constraint elicitation</td>
<td></td>
</tr>
<tr>
<td>Add expert labels</td>
<td></td>
<td>Metric learning, Human representations</td>
<td></td>
</tr>
<tr>
<td>Passive observation</td>
<td></td>
<td>Collecting contextual information</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Generative factors, concept</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>representations, Feature attributions</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

We provide this form of feedback...

During and after the model is trained.

Keys of Human-in-the-loop NLP


Allow humans to easily provide feedback.

Turn nontechnical, human preferences into usable model updates.

Build models to effectively take the feedback.
What are some forms of feedback?

- Label additional data points.
- Edit data points.
- Change data weights.
- Binary/Scaled user feedback.
- Natural language feedback.
- Code language feedback.
- Define, add, remove feature spaces.
- Directly change the objective function.
- Directly change the model parameter.
- ...
Which kinds of feedback do you prefer to provide?

- Label additional data points.
- Edit data points.
- Change data weights.
- Binary/Scaled user feedback.
- Natural language feedback.
- Code language feedback.
- Define, add, remove feature spaces.
- Directly change the objective function.
- Directly change the model parameter.
- ...
Trade-offs: Human-friendly vs. Model friendly

Models need feedback that they can respond to. Update objective function is more effective. Labeling is not as much unless large scale.

Humans prefer easier-to-provide feedback, non-experts maybe:
NL feedback > labeling > model manipulation

Experts maybe the reverse:
Because they know more about feedback effectiveness and reliable-ness.

Label additional data points.
Edit data points.
Change data weights.
Binary/Scaled user feedback.
Natural language feedback.
Code language feedback.
Define, add, remove feature spaces.
Directly change the objective function.
Directly change the model parameter.
...
Interaction Medium

Graphical user interface:
Graphic icons, visual indicators
Visualize the blackbox NLP model
Provide users more accurate control

Natural language interface:
Users interact via natural language
Explicit feedback or implicit ones
Intuitive as it simulates a conversation

What are some challenges in HITL NLP?

**Humans can only provide limited amount of feedback.**

Need to avoid cognitive overload

This is also why sometimes we may prefer local feedback, because global feedback would require a high-level understanding on the task/model which is harder to get.
What are some challenges in HITL NLP?

Humans are not oracle, and make mistakes.

Need to deal with noisy inputs, like what Snorkel is doing.
Other Open Thoughts

• As human feedback can be subjective, who should HITL systems collect feedback from? Is there any expertise levels required to perform the task?
• How to present what the model has learned and what feedback is need? How to visualize the model change after learning from user feedback?
• How to dynamically choose the most helpful feedback to collect? How to guide users to provide useful feedback?
• How to evaluate the collected human feedback as it can be noisy and even misleading?
• How to open-source tools and share user study protocols when publishing new HITL NLP work?
Exercise: Let’s build a better email assistant

Let’s divide into two groups:

**HCI:** share human insights

**NLP:** pick which solution to use