Gabriel Konar-Steenberg
K. Shankari
NREL SULI: E-mission Trip Inference System
30 June 2021

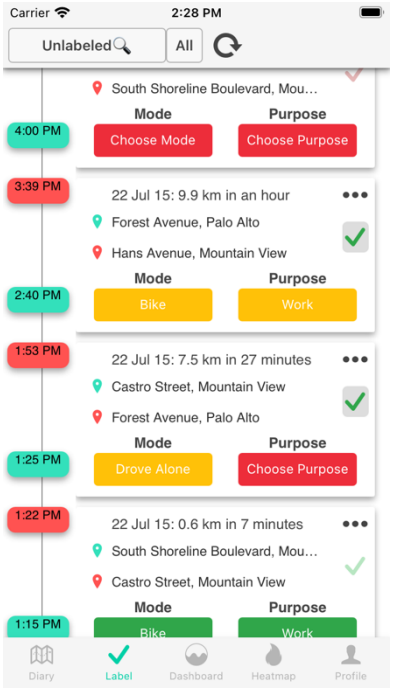
Trip Label Inference System » User Interface » Draft 2

Executive Summary

Behind the scenes, a server-side trip inference algorithm will produce a data structure that comprises, for each trip, a list of label sets with probabilities. This allows a client-side algorithm to refine the inference as the user manually confirms or corrects labels; it will also aid in analysis when the user has not confirmed or corrected labels. To the right is an example of what this data structure might look like.

```
[

],
[
    {"labels": {"mode_confirm": "bike", "purpose_confirm": "work"}, "p": 0.8},
    {"labels": {"mode_confirm": "walk", "purpose_confirm": "shopping"}, "p": 0.2}
],
[
    {"labels": {"mode_confirm": "drove_alone"}, "p": 0.8},
],
[
    {"labels": {"mode_confirm": "walk", "purpose_confirm": "shopping"}, "p": 0.45},
    {"labels": {"mode_confirm": "walk", "purpose_confirm": "entertainment"}, "p": 0.35},
    {"labels": {"mode_confirm": "drove_alone", "purpose_confirm": "work"}, "p": 0.15},
    {"labels": {"mode_confirm": "shared_ride", "purpose_confirm": "work"}, "p": 0.05}
],
```

To integrate the inference system into the phone app user interface, we will make three main changes. First, we display unfilled labels in red, inferred labels in yellow, and manually filled-out or verified labels in green. Second, we add a `To Label` view on the `Label` screen that displays only trips that users are expected to manually provide input on, according to the criteria below. Third, we change when we notify users as detailed below. We will also make some smaller UI changes to improve usability, including adding a confirm button, a feature to label many of the same type of trip at once, and a map of trips. To the right is a screenshot of progress so far, showing the red/yellow/green color scheme and the confirm button.



Expectations and notification behavior will be highly configurable to support many use cases. The basic idea of this configuration is that for each label category, comprising red labels and yellow labels with varying degrees of certainty, trip administrators will be able to select an expectation setting and a notification setting. The user could be expected to label every trip in a given category, none of them, or some random sample in between. The user could be notified after every trip in a given category, at the end of the day, or less frequently.

Study administrators will also be able to configure a “primary mode,” in which user input expectations are high, and a “secondary mode,” in which less is demanded of the user; the app can automatically cycle between primary and secondary modes according to a configurable schedule.

Please see the full proposal for a complete configuration example and many more details. Please send me any feedback you may have!

User Patterns

When thinking about the design of the system that will attempt to "autofill" trip labels to reduce the amount of user interaction required, it is useful to contemplate a number of transportation patterns. Some I have found useful to think about are:

Pattern 1. Carpooling office worker: User drives to and from work at a predictable time using a predictable route every day; however, 51% of the trips are driving alone and 49% of the time is carpooling, i.e., driving with others.

Pattern 2. Carpooling, biking office worker: Like Pattern 1 except 35% driving alone, 33% carpooling, 31% biking.

Pattern 3. Dog walker: Every evening beginning between 7pm and 8pm, user walks their dog between 0.5 and 2 miles around the neighborhood, starting and ending at their house.

Pattern 4. Plumber: Every weekday between 8am and 5pm, user makes a number of trips, the first of which is from their house and the last of which is to their house, for which the labels are `mode=drove_alone, replaced=no_travel, purpose=work`, except the last one is `purpose=home`.

Of course, we should keep in mind that all results point to our only being able to infer labels for a limited number of trips for each user, and that the less regular a user's transportation usage is, the less we will be able to predict — the examples above are towards the easier to predict end of the spectrum.

Inference Format

The above user patterns make it clear that the inference system will sometimes be able to predict one of a trip's labels with a high degree of certainty but have some trouble with the others. For instance, for the carpooling office worker, the system can quite easily learn which trips have `purpose=work`, but it will be hard to distinguish carpooling from driving alone. However, once the user inputs `mode=drove_alone`, we can infer `replaced=shared_ride`. Thus, behind the scenes, we will build an inference system that gives the phone app a list of possible label combinations and their probabilities in such a way that we will be able to calculate the most likely value of a label given information about the other labels on the trip and the other things we know about the trip. We will also design this inference system to be flexible enough to accommodate more labels than the three per trip we have been using by default. This inference system will also allow us to extrapolate nicely when the user does not manually confirm inferred labels: for instance, for the carpooling office worker, if we had to infer a single mode to ask the user to confirm, we would say `drove_alone` as that is technically the majority, but if the user does not confirm this we can assign 51% of the trips `drove_alone` and 49% of them `shared_ride` for analysis of results.

The screenshot below shows some sample inference results to be transmitted to the user's phone for a study that only tracks `mode` and `purpose`. Each top-level entry represents a trip; each entry within that represents possible label tuples for that trip. The first trip is empty: we have no idea what the labels are. For the second trip, we are 80% sure that the user biked to work and 20% sure that they walked to the store. To begin with, we would display the labels `mode=bike`, `purpose=work`, but if the user inputs `mode=walk`, the phone can infer `purpose=shopping`. For the third trip, we have some idea about the mode but no idea about the purpose. I have not yet decided whether this should be considered a valid inference entry. The last trip provides many options for inference that depend on what the user confirms for one of the labels.

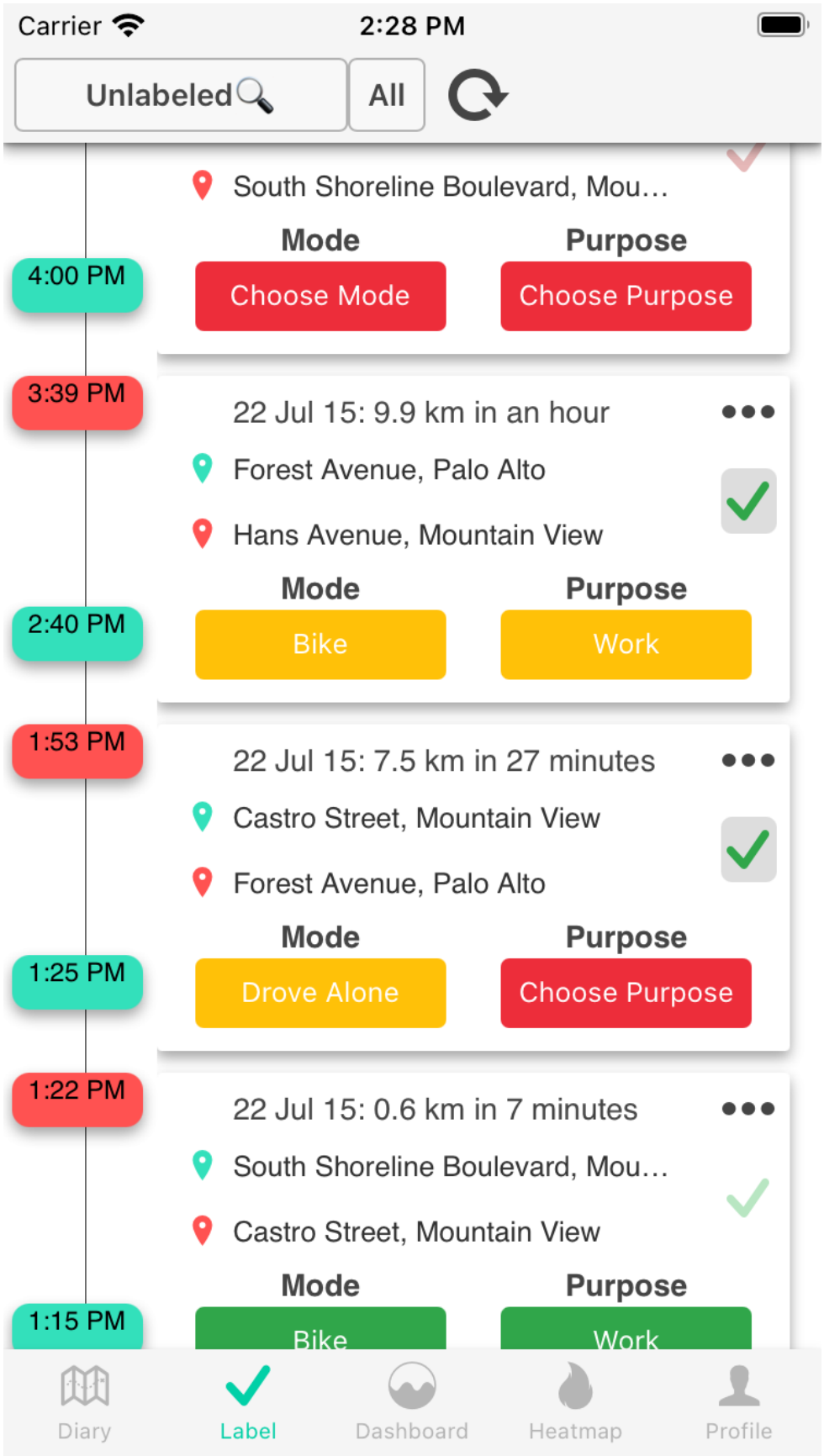

```
[

],
[
    {"labels": {"mode_confirm": "bike", "purpose_confirm": "work"}, "p": 0.8},
    {"labels": {"mode_confirm": "walk", "purpose_confirm": "shopping"}, "p": 0.2}
],
[
    {"labels": {"mode_confirm": "drove_alone"}, "p": 0.8},
],
[
    {"labels": {"mode_confirm": "walk", "purpose_confirm": "shopping"}, "p": 0.45},
    {"labels": {"mode_confirm": "walk", "purpose_confirm": "entertainment"}, "p": 0.35},
    {"labels": {"mode_confirm": "drove_alone", "purpose_confirm": "work"}, "p": 0.15},
    {"labels": {"mode_confirm": "shared_ride", "purpose_confirm": "work"}, "p": 0.05}
],
```

Major Elements of the User Interface Redesign

To implement the trip inference system, we will make the following major changes to the app's user interface:

- Instead of showing empty labels in gray and filled-out labels in green, we will show **unfilled labels in red, inferred labels in yellow, and manually filled-out or verified labels in green**.
    - This provides an easy way for the user to see what they need to do.
    - The red/yellow/green palette is intuitive and fits with the transportation theme.
    - Throughout this document, I will refer to red/yellow/green labels with these definitions in mind.
- For each trip on the `Label` screen, there will be a small checkmark `confirm` button; this button turns all of the yellow labels for that trip green. Users can also confirm individual fields by manually selecting them and choosing the option from the drop-down menu as before.
- Currently, the `Label` screen has three views: `Unlabeled`, `Invalid`, and `All`. **Add a fourth, `To Label`, and make this the default.** The `To Label` view displays trips that we expect a user to label (as detailed below) from when they are recognized to when the user labels them.
    - **This is one of the major elements of the new scheme to get the user to pay attention to exactly what they need to pay attention to, no more, no less.**
- Add a `Batch label` button on the `Label` screen. This lets the user select multiple trips and apply the same set of labels to all of them.
    - Maybe do some tests of whether users actually use this feature.
- Add a button on the `Diary` screen to show a map of all trips; add options to filter by date.
    - For now, I think we should *not* display clusters of common trips on this map; we'll think of clusters as part of the back-end complicatedness that users do not need to see. Revisit in the future to see if clusters are simple and sane enough that the user might benefit from seeing them.
- Send notifications only according to the scheme below

So far, the red/yellow/green color-coding and the confirm button have been implemented; see the screenshot below.


Carrier
2:28 PM
Unlabeled
All
South Shoreline Boulevard, Mou…
Mode
Purpose
4:00 PM
Choose Mode
Choose Purpose
3:39 PM
22 Jul 15: 9.9 km in an hour
Forest Avenue, Palo Alto
Hans Avenue, Mountain View
Mode
Purpose
2:40 PM
Bike
Work
1:53 PM
22 Jul 15: 7.5 km in 27 minutes
Castro Street, Mountain View
Forest Avenue, Palo Alto
Mode
Purpose
1:25 PM
Drove Alone
Choose Purpose
1:22 PM
22 Jul 15: 0.6 km in 7 minutes
South Shoreline Boulevard, Mou…
Castro Street, Mountain View
Mode
Purpose
1:15 PM
Bike
Work
Diary
Label
Dashboard
Heatmap
Profile

Expectations and Notifications

The rest of the UI redesign consists of clearly communicating what the user is expected to label and effectively timing push notifications to remind them to do this. The basic idea here is that trips that users are expected to label appear on the "To Label" screen, that notifications are sent at configurable times to prompt users to label these trips, and that users are provided with a textual description of what kinds of trips they will be expected/prompted to label. Though different types of labels might have different notification settings (e.g., one might configure the app to notify users about red labels every day but yellow labels only every week), the desired behavior is that whenever a certain event triggers a label notification, the user labels all of the trips on the "To Label" screen.

Different use cases might have different needs when it comes to expectations and notifications. Thus, I've designed a highly flexible set of configuration options for individual study administrators to choose from. Andy Duvall suggested that studies might want to have a primary data collection period — a short time frame every so often in which users are prompted to label more or less everything — and a secondary data collection period, the rest of the time, in which the expectations would be lowered; this is a possibility in the configuration options as well.

Expectation/Notification Configuration

These configuration options are filled out for each study by those running the study. The basic idea is that for each collection mode, the study may configure:

1. whether we expect the user to manually label various types of red and yellow labels, and
2. how often we should notify the user to do that labeling.

To implement this, for each collection mode, the study may configure a set of rules. Each rule has three parts: a trigger, which specifies the type of label that activates the rule, an expectation, which specifies whether or not the user is expected to manually input/confirm the label, and a notify setting, which specifies when a notification should be sent. The exact options are detailed below. Boldface denotes the name of a configuration question, italics indicate options, and square brackets indicate the specified kind of input.

Enable secondary collection mode?

- *[yes/no]*

If secondary collection mode is enabled:

- **Primary collection mode trigger:**
  - *Every [input number] [days/weeks/months] starting on [date]*
- **Primary collection mode duration:**
  - *[input number] days*

For each collection mode (i.e., if secondary collection mode is enabled, have two versions of the below settings), we can configure how confident the system needs to be in an inferred label before displaying it instead of keeping the label red:
Inference confidence threshold: *[input number]%*

Finally, for each collection mode we have a list of rules. There must be a rule triggered by red labels, there must be a rule triggered by yellow labels, and there can optionally be additional rules triggered by various confidence levels on yellow labels. The thinking here is that we might ask for user input for yellow labels only when we are not very confident in our inference. Rules look like this:

- **Trigger:** (the type of label that the rule operates on. If a given label falls under multiple triggers, the most specific trigger governs.)
  - *Red labels*
  - *Yellow labels*
  - *Yellow labels for which our confidence in our inference is ≤ 99%*
  - *Yellow labels, confidence ≤ 95%*
  - *Yellow labels, confidence ≤ 90%*
  - *Yellow labels, confidence ≤ 85%*
  - *Yellow labels, confidence ≤ 80%*
  - *Yellow labels, confidence ≤ 75%*
  - *Yellow labels, confidence ≤ 50%*

    - *Yellow labels, confidence ≤ 25%*
    - *Yellow labels, confidence ≤ 10%*
- **Expectation:** (whether or not the user is expected to label, and whether or not we consider sending a notification. If not expected, the user may still choose to label.)
    - *Label* (trip appears in To Label)
    - *Label random [input number]% of trips* (randomly selected trips appear in To Label)
    - *Label random [input number] days per week* (if multiple rules use this, we try to overlap days)
    - *No label* (trip does not appear in To Label)
- **Notify:** (when to notify the user, if at all. If Expectation is "No label," Notify is automatically Never. Also, we only notify if there is actually a triggering trip to be labeled.)
    - *After each trip*
    - *At the end of each day*
    - *Every [input number] days*
    - *Every [input day of the week]*
    - *Never*

Example configuration

That's a lot of information, so here's an example that might be a sensible first attempt at a configuration for the upcoming Colorado pilot. Here, boldface indicates the choices that have been made, and explanations are in italics.

Enable secondary collection mode?

- **Yes**

Primary collection mode trigger:

- Every **1 months** starting on **2021-09-01**

Primary collection mode duration:

- **7** days

Translation: do primary collection mode for the first week of every month.

Primary collection mode:
Inference confidence threshold: 65%

- Trigger: Red labels
    - Expectation: **Label**
    - Notify: **End of each day**
- Trigger: Yellow labels
    - Expectation: **Label**
    - Notify: **Every Friday**
- Trigger: **Yellow labels, confidence ≤ 75%**
    - Expectation: **Label**
    - Notify: **End of each day**

The thinking here is that we want people to label novel trips frequently enough that they remember what to fill in, and yellow labels with a low enough probability should be treated as somewhat novel. For yellow labels with a higher probability, we still want user verification, but this can happen less frequently if they are the only trips we need to label. Thus, we will try to wait for red labels or very unsure yellow labels before prompting the user to verify less unsure yellow labels, but if those trips don't happen, we'll have the user verify at the end of the work week.

Secondary collection mode:
Inference confidence threshold: 55%

- Trigger: Red labels
    - Expectation: **Label random 2 days per week**
    - Notify: **End of each day**
- Trigger: Yellow labels
    - Expectation: **No label**
    - Notify: None

- Trigger: **Yellow labels, confidence ≤ 95%**
    - Expectation: **Label random 5% of trips**
    - Notify: **End of each day**
- Trigger: **Yellow labels, confidence ≤ 75%**
    - Expectation: **Label random 2 days per week**
    - Notify: **End of each day**

The thinking here is that we want to minimize user interaction as much as possible while still trying to pick up on any major shifts. Thus, we ask for labels on red trips and somewhat uncertain yellow trips twice a week and do spot checks on yellow trips we aren't extremely sure of. Note that even though we have all the notify settings at "End of each day," this does not *mean the user will get a notification at the end of each day — the user will only get a notification if the day is one of the random 2 days per week and they have certain label types or if one of the random 5% of certain trips fell on that day.*

Miscellaneous Notes

- One feature not included is the ability to nag users who ignore the first notification. I think that is okay, because subsequent notifications for new trips should serve that purpose, but we should keep an eye on whether this would be helpful.

Conclusion

Feel free to provide any feedback on this draft you might have!